UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

ESCUELA DE INGENIERÍAS INDUSTRIALES

MATEMÁTICA APLICADA

Matemática aplicada

PROYECTO FIN DE MÁSTER

MODELOS DE MACHINE LEARNING PARA EL ESTUDIO DE LA CONTAMINACIÓN Y SU IMPACTO EN LA SALUD

Máster en

SISTEMAS INTELIGENTES EN ENERGÍA Y TRANSPORTE

Autor: CARMEN SERRANO PORTILLO

Tutor: GABRIEL AGUILERA VENEGAS

Cotutor JOSE LUIS GALAN GARCIA

MÁLAGA, junio de 2.025

A mi director de proyecto, Gabriel Aguilera, por su constante orientación, compromiso y apoyo a lo largo de este trabajo.

A mi familia y a mi pareja, por su infinita paciencia, comprensión y ánimo incondicional durante todo el proceso.

Y, por último, a mis compañeros, por haber enriquecido mi formación con su compañía, colaboración y experiencias compartidas durante estos años.

Resumen

En el contexto actual, los debates sobre el cambio climático han reavivado la preocupación acerca de los efectos de la contaminación ambiental en nuestro entorno. En este proyecto se plantea un enfoque innovador mediante el uso de técnicas de aprendizaje automático (Machine Learning) para analizar el impacto de la contaminación ambiental en la salud pública.

El trabajo actual se basa en la comparación de diversos métodos automáticos de análisis, con especial énfasis en los bosques aleatorios (*Random Forests*), destacando su potencial para modelar relaciones complejas en datos ambientales y de salud.

Para llevar a cabo el estudio, se ha realizado un exhaustivo preprocesamiento de datos, así como una evaluación rigurosa de métodos de interpolación espacial, priorizando el método de ponderación inversa por distancia (*Inverse Distance Weighting*) por su efectividad y adecuación al tipo de datos analizados.

Este proyecto contribuye a ampliar la perspectiva bioestadística tradicional, integrando herramientas avanzadas de Machine Learning para ofrecer una interpretación más precisa y robusta del impacto de la contaminación en la salud, facilitando así la toma de decisiones informadas en políticas públicas ambientales y sanitarias.

Palabras clave: contaminación, salud, *Machine Learning*, interpolación espacial, bosques aleatorios, bioestadística.

Abstract

In the current context, climate change debates have renewed concern about the effects of environmental pollution on our surroundings. This project proposes an innovative approach using machine learning techniques to analyze the impact of environmental pollution on public health.

The present study is based on the comparison of various automated analysis methods, with special emphasis on Random Forests, highlighting their potential to model complex relationships in environmental and health data.

To this end, an exhaustive data preprocessing has been carried out, along with a rigorous evaluation of spatial interpolation methods, prioritizing the Inverse Distance Weighting method for its effectiveness and suitability to the type of data analyzed.

This project contributes to expanding the traditional biostatistical perspective by integrating advanced machine learning tools to provide a more precise and robust interpretation of pollution's impact on health, thus facilitating informed decision-making in environmental and health public policies.

Keywords: *pollution, health, Machine Learning, spatial interpolation, random forests, biostatistics.*

Índice

Índice de figuras

Índice de tablas

1. Introducción

1.1. Preámbulo

Durante las últimas décadas el aumento de la contaminación y gases de efecto invernadero empiezan a generar preocupación a nivel internacional. Las transformaciones ambientales producidas se traducen en riesgos para la biodiversidad y ecosistemas por lo que es natural preguntarse si podría tener afectaciones también en la propia salud humana.

El crecimiento económico y la urbanización llevan también asociados el desarrollo de ciertas industrias como lo pueden ser la petrolera, los servicios, agro-alimentaria y automotriz. Éstas, traen también consigo un consumo intenso de combustibles fósiles y con ello la generación de elevados volúmenes de contaminantes [1].

Las evidencias del impacto del cambio climático sobre la salud cada día se hacen más evidentes. Siendo España uno de los países más vulnerables frente al cambio climático. Estas afectaciones en la salud se pueden apreciar en el aumento de la mortalidad por olas de calor, aumento de la contaminación por partículas finas y ozono o incluso aumento de la temperatura de mares y océanos que desemboca en fenómenos meteorológicos extremos como, por ejemplo, la DANA en Valencia. Estos eventos subrayan los riesgos, no solo para la biodiversidad y ecosistemas, sino también para la salud humana.

Por otro lado, la respuesta de la población a estos agentes contaminantes varía, hay personas que son más susceptibles y vulnerables que otras. Una suma de factores como ambiente social desfavorable, dietas inadecuadas, exposición a riesgos laborales, hábitos no saludables junto a la exposición de contaminantes peligrosos puede incrementar el riesgo de enfermar por encima de los esperado [2].

El *Machine Learning* se describe como el grupo de estrategias analíticas que tienen como propósito el desarrollo de algoritmos que permitan clasificación, exposición, predicción de los datos [3].

El uso de técnicas de *Machine Learning* es un campo en crecimiento y, entre sus aplicaciones, se encuentra la salud human. Estas técnicas se pueden aplicar desde la detección temprana de diversas enfermedades como la diabetes, linfomas y metástasis, así como otras más susceptibles a subjetividad, aquellas pertenecientes al campo de la psicología y psiquiatría [4].

1.2. Antecedentes

Se estima que en países industrializados un 20 % de las enfermedades se pueden atribuir a causas medioambientales (contaminación de aire externo e interno, aguas, sustancias y preparados químicos, etc.) [2].

Numerosos estudios se han enfocado en demostrar que enfermedades como el cáncer pueden estar relacionados con la contaminación, se ha estimado que entre el 1-2 % de casos de cáncer están asociados con la presencia de una gran concentración de gases y compuestos como benzo(a)pireno, benceno, algunos metales, partículas finas y posiblemente, ozono [5].

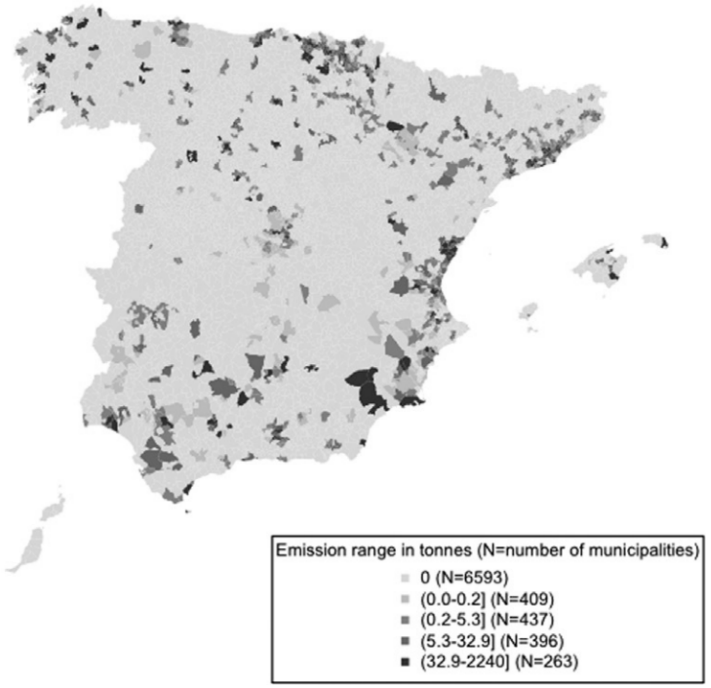


Figura 1: Cantidad total de emisiones de sustancias cancerígenas clasificadas como Grupo 1 por la IARC (2010). Referencias: [5].

La *International Agency of Research on Cancer (IARC)* ha clasificado los agentes en función de su potencial cancerígeno, se considera el Grupo 1 aquellos agentes que son cancerígenos para los humanos [6]. En la Fig.1 se muestran las zonas más expuestas a este tipo de sustancias donde se puede observar que una gran cantidad de emisiones se encuentran cercanas a los núcleos urbanos y además se expande por todo el territorio español [5].

El estudio [5] sugiere que se aquellas personas que residen en ciudades situadas cerca de fuentes de contaminación tienen un mayor riesgo de sufrir muertes por enfermedades como el cáncer que aquellas que viven en zonas no industrializadas. También sustancias de origen urbano como el dióxido de nitrógeno está relacionados con un aumento del riesgo a corto plazo de hospitalizaciones o mortalidad por causas cardiovasculares y respiratorias [7].

Las zonas más industrializadas y por tanto afectadas por agentes contaminantes son las regiones de Aragón (24 %), Andalucía (17 %) y Cataluña (15 %), entre éstas industrias se encuentran instalaciones dedicadas a la ganadería y fábricas de plásticos, asfalto, cemento, textiles, cerámicas, etc. [8].

Actualmente el *Machine Learning* están siendo usado de forma extensiva en la predicción de concentraciones de partículas pero también para asignación de fuentes y la salud humana. El artículo [9] plantea la posibilidad de que este tipo de estudios requiera de datos de más calidad ya que el ML enfrenta desafíos como la falta de datos específicos, la limitación de modelos para tareas complejas y la ignorancia de factores antropogénicos.

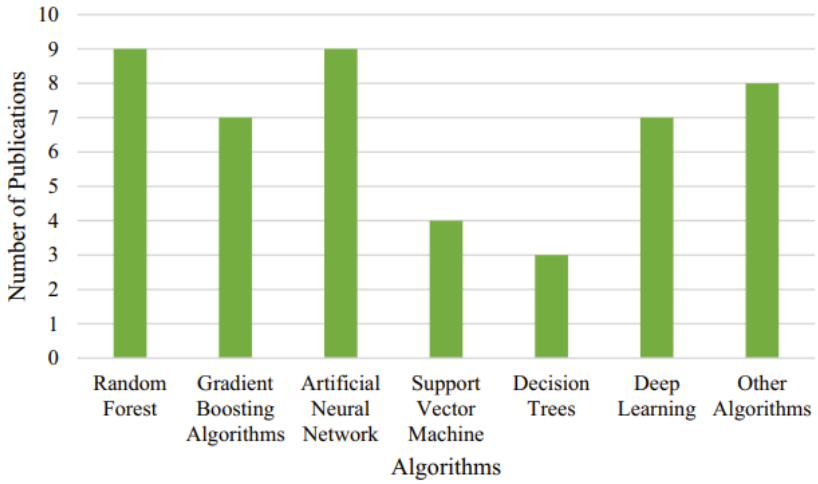


Figura 2: Distribución de modelos de *Machine Learning* en artículos publicados. Referencias: [10].

En la figura 2 se muestra el uso de los diferentes algoritmos en artículos publicados donde se puede ver predominancia del uso de *Random Forest* y *Neural Network* frente al resto. El artículo [10] indica que hay tres clases de algoritmos de *Machine Learning* que tienen un rendimiento predictivo muy alto: *Random Forest*, *Gradient Boosting Machine* y árboles de clasificación y regresión (CART).

1.3. Objetivos e hipótesis

El presente proyecto tiene como propósito explorar el impacto de la contaminación en la salud humana haciendo uso de herramientas de *Machine Learning*. Para ello, se han establecido una serie de objetivos que guiarán el desarrollo del proyecto y garantizarán un enfoque estructurado y científico con el fin de obtener conclusiones significativas y aplicables dentro del ámbito académico.

1.3.1. Objetivo general

El objetivo general del presente proyecto será el desarrollo mediante diferentes técnicas de *Machine Learning*, con énfasis en árboles de decisión, de un algoritmo que tiene como fin analizar y predecir el impacto de la contaminación en la salud de la población, contribuyendo a una mejor comprensión de los riesgos asociados y a la toma de decisiones informadas en políticas de salud pública.

1.3.2. Objetivos específicos

Los objetivos específicos serán los siguientes:

- Realizar una investigación de la literatura científica con el fin de identificar patrones, establecer variables clave y plantear las primeras hipótesis.
- Completar y enriquecer la base de datos con información de contaminación ambiental, utilizando métodos de interpolación espacial para estimar los valores en zonas sin datos directos, y desarrollando un programa que automatice esta tarea. Finalmente, se realizará una limpieza de la base de datos para optimizar su uso en el programa.
- Implementar un modelo basado en árboles de decisión que permita evaluar el impacto de la contaminación en la salud humana.
- Investigar y desarrollar algoritmos adicionales haciendo uso de distintas herramientas de *Machine Learning* e interpolación para la posterior evaluación del desempeño, interpretabilidad y tiempo de ejecución de cada herramienta.
- Analizar los diferentes resultados obtenidos con el fin de obtener conclusiones respecto a la afectación de la contaminación en la salud y el mejor modelo de

Machine Learning para la obtención de resultados fiables.

1.3.3. Hipótesis

En el siguiente apartado se presentarán las hipótesis tras una evaluación de la literatura científica. Los estudios revisados sugieren que la exposición a ciertos contaminantes pueden tener afectaciones en la salud y además, se muestra un creciente interés en la aplicación de técnicas de *Machine Learning* con el fin de predecir y evaluar estos impactos. A partir de esta base, se plantean las siguientes hipótesis:

- **Hipótesis 1:** "Las concentraciones de agentes contaminantes están positivamente relacionados con una incidencia en la salud humana, especialmente enfermedades respiratorias y cardiovasculares."
  - **Fundamentación:** Estudios previos muestran una relación de ciertos agentes contaminantes con un aumento de enfermedades respiratorias y cardiovasculares [2][5][7].
  - **Método de evaluación:** Se hará uso de modelos de *Machine Learning* para identificar la correlación entre agentes contaminantes y alteraciones en parámetros de la salud.
  - **Resultados esperados:** Se espera la confirmación de la Hipótesis 1, es decir, que en zonas más expuestas a la contaminación haya una alteración de parámetros de la salud. Este hallazgo contribuiría a un mejor entendimiento de riesgos asociados con el fin de poder prevenir e informar.
- **Hipótesis 2:** ."El modelo de *Machine Learning* que presenta un mayor rendimiento en la evaluación de impactos en la salud por contaminación será *Random Forest*"
  - **Fundamentación:** Basada en el artículo [10], donde se destaca un desempeño superior de ciertos algoritmos para las tareas de evaluación de contaminación y salud.
  - **Método de evaluación:** Comparación sistemática de distintos modelos en términos de precisión, sensibilidad y fiabilidad.
  - **Resultados esperados:** Justificación de elección del algoritmo más adecuado con el fin de informar para futuros estudios, optimizando su uso en aplicaciones prácticas.

Hipótesis 3: .[El] uso de técnicas de interpolación espacial, como *IDW (Inverse Distance Weighting)*, permite estimar con fiabilidad los valores de contaminación en zonas sin datos directos, mejorando la cobertura y utilidad del conjunto de datos"

- **Fundamentación:** Apoyada en estudios como [11], donde se destaca la eficacia de métodos de interpolación espacial en la estimación de variables ambientales en regiones con escasa instrumentación.
- **Método de evaluación:** Análisis comparativo entre los datos interpolados y los datos reales (cuando estén disponibles), así como evaluación del impacto en el rendimiento de los modelos de Machine Learning al incluir estos valores estimados.
- **Resultados esperados:** Validación de la interpolación como técnica útil para enriquecer la base de datos, facilitando un análisis más amplio y robusto sobre la relación entre contaminación y salud.

1.4. Plan de trabajo

El siguiente plan de trabajo tiene como objetivo detallar las actividades y plazos necesarios para la realización del presente proyecto de investigación científica. Se garantizará el cumplimiento de los objetivos de forma efectiva, también la utilización de recursos y tiempo de manera óptima.

- **Revisión bibliográfica:** donde se realizará una revisión de la literatura científica con el fin de formular conclusiones e hipótesis que se pondrán a prueba a lo largo del presente proyecto.
  - Búsqueda de artículos científicos que se encuentren relacionados con el tema del presente proyecto.
  - Resumen y análisis de los puntos tratados en la literatura seleccionada que resulten de interés para la consecución de los objetivos y el desarrollo de las hipótesis.
  - Identificación de metodologías utilizadas en investigaciones anteriores sobre la incidencia de agentes contaminantes en la salud pública.

  - Formulación de hipótesis basadas en las conclusiones obtenidas de la revisión bibliográfica.

- **Obtención de datos de contaminación precisos:** se aplicarán métodos con el fin de obtener datos de contaminación en puntos concretos del mapa español.

  - Estudio de la base de datos con la que se cuenta para la obtención de datos de contaminación en España.
  - Procesamiento y organización de los datos con el fin de integrarlos en la base de datos.
  - Validación de la precisión de la cobertura geográfica y realización de un programa de interpolación para obtener datos donde la precisión no lo permita.

- **Recolección y preparación de los datos:** se llevará a cabo una ingesta de datos y un breve análisis de estos con el fin de poder trabajar con ellos correctamente y garantizar calidad y fiabilidad de los datos.

  - Revisión y limpieza de la base de datos con la cual se va a trabajar eliminando valores atípicos, realizando un manejo de datos faltantes, etc.
  - Normalización y transformación de los datos con el fin de que queden listos para el análisis como la realización de una codificación de las variables categóricas.
  - Validación de la calidad y fiabilidad de la base de datos, asegurando su consistencia para el posterior trabajo con ésta.

- **Estudio de los modelos de *Machine Learning:*** se estudiarán los distintos modelos de *Machine Learning* con sus posterior evaluación de características, ventajas y desventajas de cada uno de los algoritmos. Se considerará además su aplicabilidad al conjunto de datos disponibles.

  - Investigación mediante la revisión de literatura científica de los modelos de *Machine Learning* más adecuados.
  - Análisis de ventajas y desventajas de cada uno de los algoritmos a estudiar con el tipo de información de la que se dispone en la base de datos.

  - Elección final de los modelos que se estudiarán en el presente proyecto basados en los requisitos de éste.

- **Aplicación los modelos estudiados a las bases de datos:** se implementarán los modelos seleccionados llevando a cabo un proceso de entrenamiento y validación de éstos.

  - Preparación de los datos con el fin de realizar un entrenamiento de los modelos (*train/test split*, creación de variables predictivas, etc.)
  - Implementación y entrenamiento de los distintos modelos seleccionados haciendo uso de las bases de datos procesadas.
  - Evaluación de los modelos y ajuste de hiperparámetros si estos lo requieren.
  - Comparación de los modelos entrenados y selección de aquel modelo que presente un mejor rendimiento y fiabilidad frente a los datos presentados.

- **Análisis de los resultados y conclusiones:** se obtendrá un análisis de los resultados obtenidos y se elaborarán conclusiones que validen las hipótesis y proporcionen recomendaciones para futuras investigaciones.

  - Interpretación de los resultados obtenidos y análisis de cómo estos se relacionan con las hipótesis que se plantearon al inicio del proyecto.
  - Formulación de conclusiones y grado de consecución de los objetivos presentados.
  - Desarrollo de una serie de propuestas para futuras investigaciones en el área.

2. Estado del arte

La investigación sobre las posibles afectaciones de agentes contaminantes en la salud humana gana relevancia debido al aumento de su presencia en los entornos. Este fenómeno viene determinado por el proceso de urbanización e industrialización, junto con el uso indiscriminado de productos químicos en distintas actividades humanas.

El uso de *Machine Learning* permite abordar desafíos significativos en el campo de la salud ambiental, como la modelización de relaciones no lineales entre distintas variables. En particular, estos modelos se han aplicado en áreas como la estimación de la calidad del aire, la identificación de fuentes de contaminación, y la evaluación de sus efectos en poblaciones vulnerables.

Diversas investigaciones han abordado el tema desde distintas perspectivas, utilizando diferentes técnicas y metodologías. El objetivo de este apartado es la revisión de la literatura científica con el fin de analizar los principales estudios que abordan el tema, identificando tendencias actuales, las diferentes conclusiones, comparando enfoques metodológicos y destacando áreas que requieran de mayor investigación. A través de esta revisión se buscará establecer una base sólida para la investigación que aquí se presenta.

2.1. Contaminación y Salud Humana

2.1.1. Evidencias y tendencias

La contaminación atmosférica viene acompañando al ser humano desde hace prácticamente 500 años. El primer caso registrado de los efectos de ésta sucedió en Londres en 1952, donde los altos niveles de contaminación produjeron un aumento del número de muertes (en torno a 4000) [12]

Los contaminantes del aire incluyen contaminantes gaseosos y partículas (PM), la virulencia de estas partículas está determinada por su tamaño, composición, origen, solubilidad y su habilidad para reaccionar con el oxígeno. Algunos estudios determinan que la contaminación ambiental o neblinas tóxicas están producidas por una gran cantidad de partículas finas (partículas menores o iguales a 2,5 μm, es decir, $PM_{2.5}$) o aerosoles. También se ha descubierto que las partículas con un diámetro

menor a 10 μm tienen un mayor impacto en la salud humana [13]

La contaminación del aire es actualmente uno de los agentes que provoca mayor mortalidad. Ésta, en 2015, causó 6,4 millones de muertes en todo el mundo (2,8 millones por contaminación en el aire del hogar y 4,2 millones por la contaminación en el ambiente). En el mismo año el tabaco causó 7 millones de muertes, el VIH 1,2 millones y la tuberculosis 1,1 millones. Sin control, se espera que la contaminación cause en 2060 entre 6 y 9 millones de muertes anuales [14].

Por otro lado, la contaminación del aire fue responsable en 2015 del 19 % de muertes cardiovasculares en todo el mundo, del 24 % de muertes por enfermedades del corazón, del 21 % de muertes por accidentes cerebro-vasculares y del 23 % de muertes por cáncer de pulmón [14].

Las áreas más afectadas por la contaminación suelen ser grandes ciudades con una alta densidad de población, especialmente en el sur y este de Asia (Figura 4) y, aunque estas son las más expuestas, los efectos sobre la salud afectan a nivel mundial (Figura 3),incluyendo incluso poblaciones rurales [15].

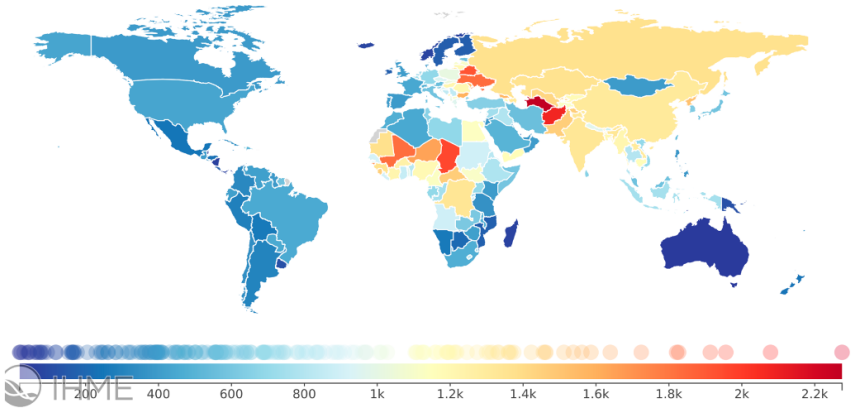


Figura 3: Carga sanitaria total en 2013 atribuida a la exposición a contaminación por $PM_{2,5}$ en el aire ambiente. Referencias: [15]

Numerosos estudios han demostrado que existe una relación significativa entre las partículas finas de contaminantes ($P_{2.5}$) y la prevalencia de enfermedades respiratorias y mortalidad. En la Unión Europea se considera que el $PM_{2.5}$ ha disminuido la media de esperanza de vida en 8.9 meses. Además, en otros estudios sobre la UE se ha encontrado que la mortalidad por problemas respiratorios se ha incrementado un 0,58 % a causa de el PM_{10} [13].

Aunque estudios pasados se han centrado en la afectación de diferentes conta-

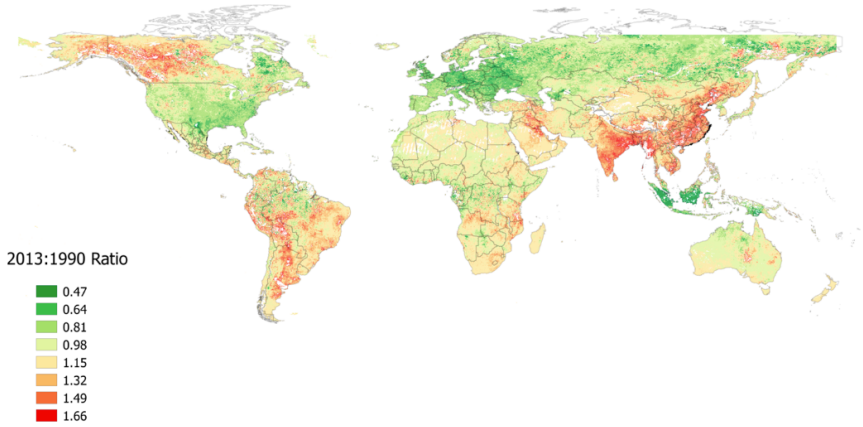


Figura 4: Ratio de la media anual de concentraciones de $PM_{2.5}$ en 2013 respecto a 1990 (rojo: aumento, verde: disminución). Referencias: [15]

minantes de manera individual, la comunidad epidemiológica ha reconocido que los efectos sobre la salud están más bien relacionados con la respuesta del cuerpo a distintas mezclas de contaminación en el aire [15].

En el artículo [16] se realizaron simulaciones de escenarios asumiendo la implementación en la Unión Europea de la legislación actualmente acordada. En la Fig.5 se puede observar la concentración de contaminación en puntos calientes o áreas urbanizadas como el norte de Italia, Polonia, Rumanía y Bulgaria. Se estima que estas áreas continuarán teniendo altas concentraciones de $PM_{2.5}$ en torno a 2030 y que por tanto se requiere de una política efectiva para reducir estos niveles.

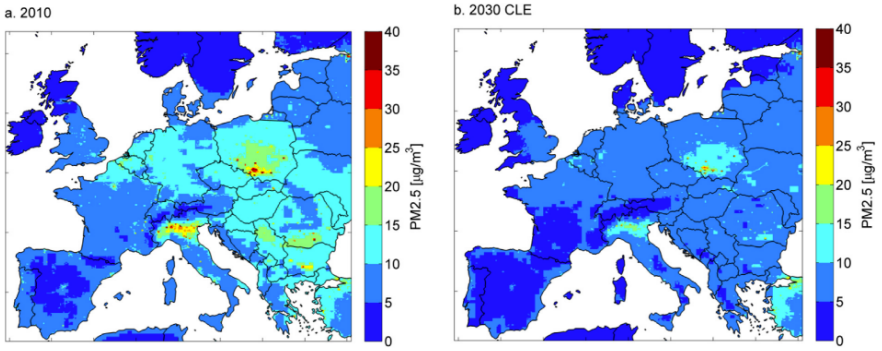


Figura 5: Simulación de la evolución de las concentraciones de $PM_{2.5}$ asumiendo la implementación de la legislación acordada entre 2010 y 2030. Referencias: [16]

Por otro lado, este mismo artículo ha simulado la pérdida de esperanza de vida para el año 2010 comparándolo con el caso de un mundo completamente limpio. Los resultados indican pérdidas mínimas de 1-2 meses en áreas como Suecia y Escocia

y máximas de 1.5 años en regiones como Polonia, Rumanía y Bulgaria [16].

Por otro lado también se estima que el 60 % de pérdida de esperanza de vida está atribuida a aerosoles secundarios formados en la atmósfera por gases precursores y el 40 % restante proviene de partículas $PM_{2.5}$. También se indica un aumento de la esperanza de vida entre 2.3 a 6 meses si se implementan correctamente las medidas legislativas para 2030 [16].

2.1.2. Los principales agentes contaminantes del aire

La concentración de contaminantes exteriores depende de una compleja interacción de varios factores que afecta a la introducción, dispersión y retirada de los contaminantes [17]:

- Tipo, naturaleza y número de fuentes.
- Características de uso de la fuente.
- Características del edificio.
- Tasas de infiltración y ventilación.
- Mezcla de aire entre y dentro de los compartimentos de un espacio interior.
- Tasas de retirada y potencial reemisión o generación por las superficies interiores y transformaciones químicas.
- Existencia y efectividad de sistemas de retiradas del aire contaminado.
- Concentraciones en el exterior

Como se ha comentado anteriormente los agentes contaminantes pueden ser bien contaminantes de tipo gaseoso o aerosoles y partículas en suspensión. Cada uno de estos agentes posee características concretas que determinan su capacidad de generar efectos adversos en la salud humana. En este apartado se describirán los principales agentes contaminantes del aire.

- **Partículas**: las partículas en suspensión son una mezcla de muchas subclases de contaminantes. Los efectos pueden ser influenciados por la composición química y su tamaño. Éstas son producidas por combustiones incompletas de

combustibles fósiles. La mayor parte de componentes de las PM son: sulfatos, nitratos, amonio, cloruros y carbón.

Por esta razón las regulaciones se centran en el diámetro de las partículas que se consideran PM_{10} cuando éste es menor a $10[\mu m]$ y $PM_{2.5}$ cuando es menor a $2.5[\mu m]$. La concentración de estas partículas se suele representar en $[\mu g/m^3]$ [18].

- **Aerosoles:** estos se pueden dividir en [18] [19]:
  - **Ozono (O_3):** formado por la acción de la luz solar en los óxidos de nitrógeno. Éste tiene variaciones en función de los momentos del día y las estaciones.
  - **Dióxido de nitrógeno (NO_2):** la mayor exposición a este tipo de compuestos se suele producir en las estufas de gas domésticas, es decir, se trata de un contaminante del hogar.
  - **Monóxido de carbono (CO):** gas tóxico producido por la combustión incompleta de combustibles carbonados como madera, petróleo, carbón vegetal, gas natural y queroseno.
  - **Dióxido de azufre (SO_2)**: resultado de la quema de combustibles fósiles (carbón y petróleo) y la fundición de menas que contengan azufre.

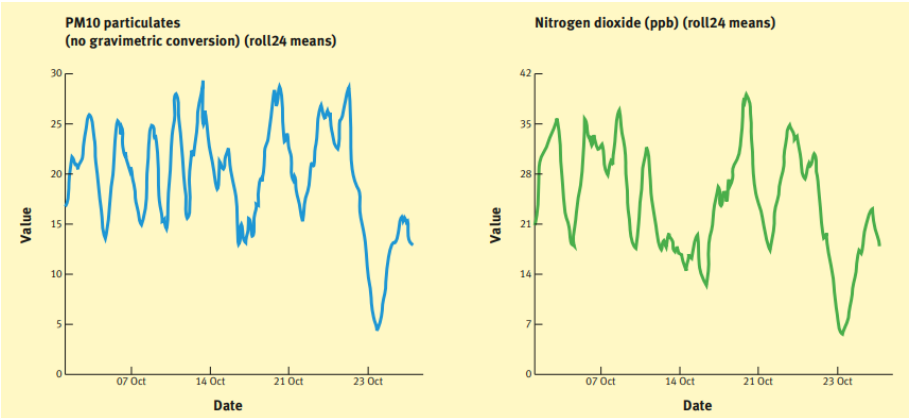


Figura 6: Media diaria de partículas PM_{10} y NO_2 medidas en Londres. Referencias:[18]

En la Fig.6 se muestran las concentraciones promedio de 24 [h!] de partículas PM_{10} y dióxido de nitrógeno (NO_2) registradas en la estación de monitoreo de *London Road, Leicester*, durante 2007.Los datos muestran cómo ambos contaminantes

tienden a presentar una evolución conjunta a lo largo del tiempo lo cual puede dar a entender que comparten fuentes comunes, como por ejemplo, el tráfico de vehículos o la quema de combustibles fósiles.

Por otro lado, la *International Agency for Research on Cancer (IARC)* expone entre los 132 agentes del Grupo 1 (cancerígenos para los humanos) una serie de agentes presentes en el exterior que pueden afectar también a la salud humana. Sin incluir los ya comentados se presentan estos otros [6]:

- Sustancias liberadas al ambiente
  - **Benceno:** contaminante presente en areas urbanas por emisiones de vehículos y procesos industriales.
  - **Dioxinas y policlorobifenilos (PCBs):** liberados en procesos de combustión y procesos industriales.
  - **Partículas de asbesto:** cercanas a industrias o sitios de demolición que presentan materiales con asbesto.
- Radiación ambiental
  - **Radón-222 y productos de desintegración:** gas radioactivo que se encuentra presente en el aire exterior cercano a minas o áreas ricas en uranio.
  - **Radiación solar (UVA, UVB, UVC):** puede ser más intensa en áreas con contaminación que reduce la protección atmosférica.
- Contaminantes industriales y de procesos
  - **Producción de aluminio y coque:** emisiones generadas en industrias.
  - **Polvo de sílice cristalina:** emisiones cercanas a industrias de minería o construcción.

Después se exponen además otro tipo de contaminantes (Grupo 2A y 2B) como posibles agentes cancerígenos para los humanos donde cabe destacar elementos más rurales y agrícolas, pesticidas volátiles y derivados químicos como por ejemplo el hexaclorobenceno entre otros agentes derivados de emisiones de la manufactura de la madera, metales y compuestos metálicos, etc. [6]

Estos datos poseen relevancia ya que permiten comprender que, aunque la prevalencia de compuestos cancerígenos cerca de zonas industriales es clara, también se pueden encontrar problemas relacionados con la contaminación exterior en zonas rurales y mineras entre otras.

La Organización Mundial de la Salud (OMS) identifica las principales fuentes de emisión y exposición, destacando que el principal elemento de la contaminación es la combustión de combustibles fósiles y biomasa para la generación de energía. Esto también puede ocurrir en espacios interiores debido al uso de elementos como estufas, la combustión de tabaco, entre otros [19].

Otras fuentes de combustión exterior, dispersas por tierra, aire y agua incluyen la industria y la generación de energía, la quema de biomasa (ya sean fuegos controlados o no, de bosques y residuos agrícolas o urbanos), así como la re-suspensión de polvo superficial y actividades de la construcción [19].

La concentración temporal de contaminantes exteriores varía en función de la distribución de las fuentes y el patrón de funcionamiento (diario, estacional, entre otros), las características de los contaminares y su dinámica (dispersión, deposición, interacción con otros contaminantes) y las condiciones meteorológicas [19].

Cuadro 0.1. Niveles recomendados de las directrices sobre la calidad del aire y metas intermedias

Contaminante	Tiempo promedio	Meta intermedia				Nivel de las directrices sobre la calidad del aire
		1	2	3	4	
$MP_{2,5}$, µg/m³	Anual	35	25	15	10	5
	24 horas[a]	75	50	37,5	25	15
MP_{10}, µg/m³	Anual	70	50	30	20	15
	24 horas[a]	150	100	75	50	45
O_3, µg/m³	Temporada alta[b]	100	70	–	–	60
	8 horas[a]	160	120	–	–	100
NO_2, µg/m³	Anual	40	30	20	–	10
	24 horas[a]	120	50	–	–	25
SO_2, µg/m³	24 horas[a]	125	50	–	–	40
CO, mg/m³	24 horas[a]	7	–	–	–	4

[a] Percentil 99 (es decir, 3-4 días de superación por año).
[b] Promedio de las concentraciones máximas diarias de O_3 (medias octohorarias) en los seis meses consecutivos con la concentración media móvil de O_3 más alta.

Figura 7: Recomendaciones de la OMS sobre calidad del aire y metas intermedias. Referencias:[19]

En la Fig.7 se muestran las orientaciones de la OMS sobre la Calidad de Aire y límites de los principales contaminantes atmosféricos que entrañan riesgos para la salud. También establecen metas intermedias para promover la reducción de estas concentraciones y los beneficios para la salud conexos a esta reducción. A modo de ejemplo, indica la OMS, alcanzar la meta intermedia 1 (35 $[\mu g/m^3]$) permitiría evitar unas 300.000 defunciones al año en todo el mundo [19].

2.1.3. Afectación de la contaminación en grupos vulnerables

La carga de las enfermedades relacionadas con la contaminación no se encuentra igualmente distribuida, generalmente esta desproporción afecta principalmente a grupos más vulnerables y susceptibles . El impacto de la contaminación del aire puede verse en individuos con mayores niveles de exposición y en personas con enfermedades crónicas (como asma, EPOC, diabetes, insuficiencia cardíaca y cardiopatías isquémicas), así como niños y mujeres embarazadas [19].

Los fetos y niños, en comparación con los adultos presentan una vulnerabilidad especial a los tóxicos ambientales debido a su inmadurez fisiológica y menor exposición vital a estos. Además, los niños inhalan un volumen de aire relativamente mayor que los adultos y suelen pasar un mayor tiempo al aire libre [20].

La mortalidad intrauterina, perinatal o neonatal ha sido ampliamente estudiado. En 1990 se llevó a cabo un estudio en la República Checa donde encontraron una asociación entre las concentraciones de SO_2 y partículas en suspensión $PM_{2.5}$ con las tasas de mortalidad neonatal por distritos. Además también se han llevado a cabo estudios que han encontrado relaciones entre contaminación atmosférica y crecimiento fetal y otros que han evaluado la relación entre la contaminación y el retraso del crecimiento intrauterino aunque estos campos aun están por desarrollar y contrastar [20].

En la Tabla 1 se muestra una revisión bibliográfica de diferentes estudios donde se relacionan ciertas causas de mortalidad infantil con agentes contaminantes. Las sigas AOR se corresponden con *Adjusted odd ratio*, es decir, la probabilidad de que ocurra un evento (mortalidad) en el grupo expuesto frente al grupo no expuesto, si este valor es mayor que 1 el riesgo que de ocurra el evento es mayor en el primer grupo, en este caso el el AOR se calcula para cada incremento de 50 $[\mu g/m^3]$ del contaminante indicado. Por último, las sigas SE indican "Sin Efecto".

Mortalidad	Contaminante	Resultados
Postneonatal con problemas respiratorios	TSP (*Total Suspended Particles*)	AOR = 3.91 para cada incremento de 50 $\mu g/m^3$
Postneonatal con problemas respiratorios	TSP	AOR = 1.95 para cada incremento de 50 $\mu g/m^3$
	SO_2	AOR = 1.74 para cada incremento de 50 $\mu g/m^3$
	NO_2	AOR = 1.66 para cada incremento de 50 $\mu g/m^3$
Infantil postneonatal	PM_{10}	AOR = 1.10 para cada incremento de 50 $\mu g/m^3$
Respiratoria	PM_{10}	AOR = 1.40 para cada incremento de 50 $\mu g/m^3$
Súbita infantil	PM_{10}	AOR = 1.26 para cada incremento de 50 $\mu g/m^3$
Intrauterina	NO_2	Fuerte asociación
	SO_2	SE
	CO	SE
	$NO_x + SO_2 + CO$	Asociación significativa
	O_3	SE
	PM_{10}	SE
Infantil	NO_2	SE
	SO_2	SE
	CO	SE
	O_3	SE
	PM_{10}	6.9 % para cada incremento de 10 $\mu g/m^3$
Perinatal		SE

Tabla 1: Contaminación en el aire y mortalidad infantil. Referencias: adaptado de [21]

La morfogénesis de los pulmones es un proceso lento que empieza a los 4-7 meses de gestación y continúa hasta la adolescencia. Durante las primeras fases del embarazo la diferenciación y la morfogénesis pueden verse alteradas considerando que al final del embarazo se produce un deterioro de las funciones estructurales y por tanto, puede producirse un crecimiento funcional del pulmón. Las exposiciones ambientales, incluido la contaminación del aire, pueden llevar a una alteración de la alveolización y, por lo tanto, a un deterioro del desarrollo y la función pulmonar después del nacimiento [22].

Un estudio realizado en BILD (*Basel-Bern infant lung development cohort*) sobre un grupo de nacimientos evaluaron la exposición a la contaminación ambiental durante el periodo del embarazo. Para ello se realizaron mediciones de la función pulmonar a la edad de cinco semanas. Este estudio arrojó como resultado una mayor cantidad o volumen de aire por minuto y una mayor frecuencia respiratoria asociada con una exposición mayor a partículas PM_{10} lo que da a entender que hay una

relación directa entre estos dos sucesos [22].

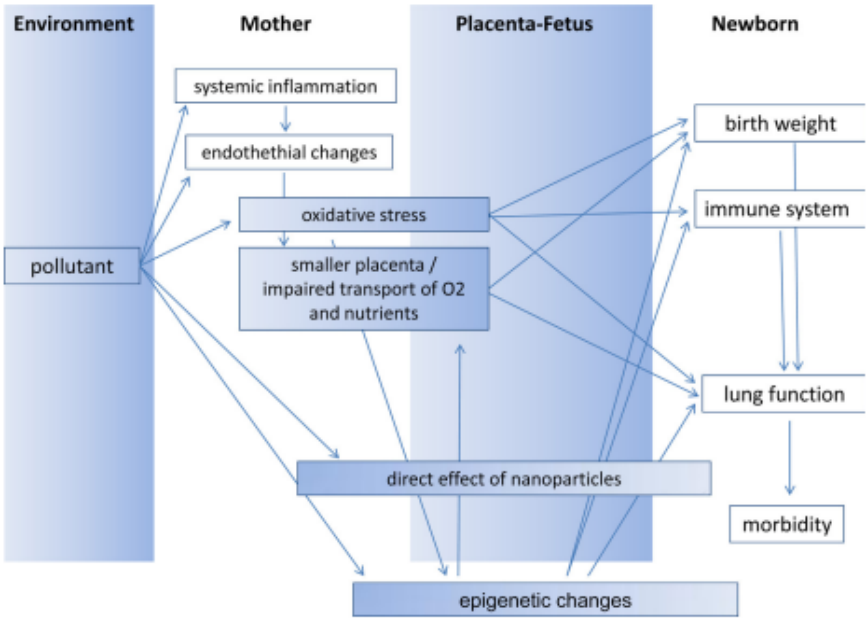


Figura 8: Causas directas e indirectas de la contaminación ambiental en el embarazo. Referencias:[22]

En la Fig.8 se muestra cómo la contaminación medioambiental afecta paso a paso en el embarazo. Esto provoca inflamaciones sistémicas y cambios en el endometrío de la madre, lo cual puede llevar a un menor tamaño de placenta y generar por tanto un menor peso en el bebé, una reducción del sistema inmune de éste o menor función pulmonar, entre otros problemas.

En la Tabla 2 se muestra un resumen de los resultados de varios estudios sobre la relación entre la contaminación ambiental y el bajo peso de los recién nacidos. De manera general, se puede observar una clara relación entre la exposición a estos contaminantes y el bajo peso de los niños. Esto sugiere que la calidad del aire podría ser un factor importante que afecta al desarrollo fetal y resalta la necesidad de controlar este tipo de sustancias.

Algunos estudios realizados en Gran Bretaña en 1971 y en Estados Unidos en 1977 indican que la mortalidad por problemas respiratorios en niños se había incrementado en regiones que sufrían de mayores problemas de contaminación [23].

Además, estudios sobre los ingresos hospitalarios en verano en el sur de Ontario (Canadá) desde 1975 hasta 1980 reportaron que los ingresos por asma en niños hasta los 14 años fueron invariablemente altos (hasta un 15 % más) después de algunos

Contaminación en el aire y mortalidad infantil		
Evento	**Contaminante**	**Resultados**
Bajo peso	SO_2 TSP	AOR = 1.21 para cada incremento de 100 $\mu g/m^3$ AOR = 1.10 para cada incremento de 100 $\mu g/m^3$
Bajo peso	TSP SO_2 NO_x	AOR = 1.04 para cada incremento de 50 $\mu g/m^3$ AOR = 1.10 para cada incremento de 50 $\mu g/m^3$ AOR = 1.07 para cada incremento de 50 $\mu g/m^3$
Bajo peso	O_3 NO_2 PM_{10} CO	SE SE SE AOR = 1.22 para CO ¿5.5 [ppm] en el primer trimestre
Muy bajo peso	TSP + SO_2	AOR = 2.88 comparando el grupo más y menos expuesto (56.7 vs 9.9 [$\mu g/m^3$])
Bajo peso	CO SO_2	AOR = 1.08 en el primer trimestre. AOR = 1.75 por incremento de 1 [ppm] en el primer trimestre AOR = 1.18/1.20 por incremento de [ppm] en todos los trimestres

Tabla 2: Contaminación del aire y bajo peso en fetos. Referencias: adaptado de [21]

días en los que el ozono en el ambiente excedía los 80 ppb en comparación con el resto de días [23]

Un estudio diferente realizado por Gauderman [24] y que contaba con 1759 niños con una media de edad de 10 años que fueron estudiados durante ocho años y además contaban con diversos ambientes. En este se encontró una reducción en el crecimiento del volumen respiratorio forzado en un segundo (FEV1) y este se encontraba relacionado con la exposición al dióxido de nitrógeno (NO_2) y al material PM_{10} pero no con el ozono (O_3).

En comunidades con alta exposición, el 7.9 % de los jóvenes de 18 años tenía una relación observada/esperada de $FEV_1 < 80\,\%$, en comparación con solo el 1.6 % en comunidades de baja exposición.

Sin embargo, otro largo estudio en Ciudad de México estudió a 3170 niños de la ciudad con altos niveles de contaminación, concluyendo que los déficit de FEV_1

tenían una gran relación con PM_{10} y el NO_2, pero también con el O_3 [25].

Los síntomas de los niños a este tipo de contaminantes son la disminución de la capacidad pulmonar, aumento de la tos e incremento de la prevalencia de asma y alergias. Ciertos artículos reportan una prevalencia de la tos sin síntomas de resfriado en grupos de 4400 niños de preescolar [18].

Por otro lado, otro grupo de riesgo serían las personas mayores. En el período de 1999-2002 se llevó a cabo un estudio que tenía como fin examinar las relaciones entre las partículas finas ($PM_{2.5}$) y los ingresos hospitalarios por problemas de corazón y pulmonares, en residentes mayores de 65 años. Este estudio demostró que cualquier pequeño incremento de las partículas finas resultaba en un incremento de los ingresos hospitalarios [26].

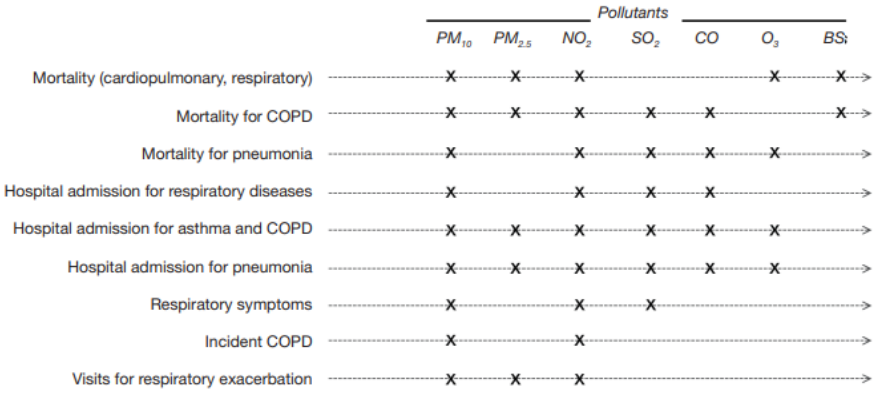


Figura 9: Principales enfermedades respiratorias producidas por contaminación exterior en mayores de 65 años en función del tipo de contaminante. Referencias:[27]

En la Fig.9 se muestra un resumen de los principales problemas de salud derivados de los agentes contaminantes exteriores más frecuentes. La mayor parte de los datos muestra mayores riesgos en la edad anciana respecto al resto de la población. Los agentes contaminantes nombrados en ésta son; monóxido de carbono (CO), dióxido de nitrógeno (NO_2), partículas importantes (PM), dióxido de azufre (SO_2), ozono (O_3), *black smoke* (BS) y por último, las siglas COPD corresponden a problemas crónicos de obstrucción pulmonar.

Los efectos de la contaminación se pueden analizar en dos contextos específicos:

- **Efectos a corto plazo:** se ha demostrado que hay una asociación directa entre la exposición a corto plazo a aire contaminado y las enfermedades respiratorias en la edad anciana.

Un estudio realizado en Barcelona [28], durante 1980 evaluó la relación entre la entrada diaria a urgencias por problemas crónicos de obstrucción pulmonar (COPD) y el aumento de dióxido de sulfuro (SO_2). Se demostró que un aumento en torno a 25 $[\mu g/m^3]$ de este compuesto en un plazo de 24 horas producía un incremento del 6 % y el 9 % de ingresos durante invierno y verano respectivamente. También se encontró esta misma relación para el *Black Smoke* durante el invierno.

Otros estudios documentados en el artículo [27] muestran una gran relación entre ingresos hospitalarios por problemas respiratorios con altos niveles de SO_2, PM_{10} y O_3 en personas mayores de 65 años. En general, la relación entre la exposición a corto plazo de aire con agentes contaminantes y la enfermedad en las personas mayores está bien documentada y cuenta con numerosa validación científica. Los efectos son principalmente COPD, asma y neumonía.

- **Efectos a largo plazo:** los efectos de la contaminación del aire a largo plazo también han sido muy estudiados y se suelen presentar con una mayor incidencia de COPD, bronquitis crónica (CB), asma y enfisema.

  En Italia entre 1980 y 1993 se llevaron a cabo dos largos estudios donde se mostró una prevalencia de síntomas respiratorios que se incrementaban con la edad y tendía a ser mayor en zonas urbanas respecto a zonas rurales. Se trataban principalmente de la tos (37 % para zonas urbanas frente a 18 % para zonas rurales), las sibilancias (39 % frente a 27 %) y el enfisema (22 % frente a 7 %) en los hombres, y para la pleuritis (32 % frente a 18 %) en las mujeres [29].

  Por otro lado, estudios en Países Bajos (1986-1994) encontraron que la proximidad a carreteras principales estaba asociada con un mayor riesgo de mortalidad. En China (1998-2009) también se encontraron asociaciones fuertes entre la exposición a largo plazo al NO_2 y PM_{10} y un mayor riesgo de mortalidad por enfermedades respiratorias en personas mayores de 60 años. Por último, otro estudio en este mismo lugar mostró que por cada aumento de 10 $[\mu g/m^3]$ de PM_{10} el riesgo de mortalidad respiratoria aumentaba en un 1,7 % [27].

2.1.4. Afectación de la contaminación en el marco español

Un estudio realizado por Ecologistas en Acción [30] analizó la calidad del aire que respiró en 2023 la población española. De éste se obtienen numerosas conclusiones; superado la crisis del COVID-19 la calidad del aire ha mejorado respecto a las partículas PM_{10} y $PM_{2.5}$. A pesar de esto un 6 % de la población se encuentra aun expuesto a aire que incumple los actuales estándares legales.

Town/City	% Days of exceedance of the WHO limit. PM_{10}	% Days of exceedance of the WHO limit. $PM_{2.5}$
Coruña	12.3%	
Albacete	35.8%	
Almería	25.8%	
Badajoz	1.2%	
Bilbao	17.1%	
Burgos	5.8%	
Cáceres	1.1%	
Córdoba	40.0%	
Cuenca	10.2%	
Granada	29.0%	
Guadalajara	10.7%	
Huelva	10.9%	
Jaén	25.0%	
León	20.9%	
Logroño	10.8%	
Madrid	13.5%	14.9%
Málaga	11.8%	
Murcia	6.3%	
Pamplona	11.3%	
Pontevedra		
Ourense	3.7%	
Oviedo	44.4%	
Palma Mallorca	3.9%	
Las Palmas	16.7%	7.9%
Salamanca	12.2%	
Sta. Cruz Ten	32.3%	7.2%
San Sebastián	8,3%	
Santander	11.9%	
Segovia	31.5%	
Sevilla	23.0%	
Soria	6.0%	
Toledo	23.3%	
Valencia	7.3%	
Valladolid	0.9%	
Vitoria	9%	
Zamora	7.3%	
Zaragoza	22.5%	

Figura 10: Porcentaje de días en el que se supera el umbral de PM_{10} y $PM_{2.5}$. Referencias:[31]

En la Fig.10 se muestra el porcentaje de días donde se supera el umbral medio diario establecido para las partículas de PM_{10} (50 $[\mu g/m^3]$) y $PM_{2.5}$ (25 $[\mu g/m^3]$). Donde se puede observar valores muy altos en las ciudades de Oviedo y Córdoba. Y, por otro lado, observando un gran porcentaje de días en los que se supera el límite de partículas $PM_{2.5}$ en Madrid, Las Palmas y Santa Cruz de Tenerife.

En general, en las regiones españolas como un todo el $PM_{2.5}$ es el contaminante

que está asociado con la mayor parte de muertes en pueblos y ciudades seguido del NO_2, el PM_{10} y el O_3 [31].

Un estudio realizado en España indica que el 70 % de las industrias químicas se encuentra en el noroeste de la península. Un total de 4768 toneladas de agentes cancerígenos fueron encontradas en aire, tierra y agua. Éstas procedían de industrias químicas inorgánicas (42 %), seguido de las orgánicas (32 %), fertilizantes (18 %), industrias farmacéuticas (4 %) y de explosivos/pirotecnia (4 %) [32].

Este mismo estudio ha encontrado una relación entre el incremento de cáncer colorrectal cerca de industrias químicas orgánicas y cáncer de ovario asociado a estas mismas zonas. Por otro lado, en zonas cercanas a industrias químicas inorgánicas y de explosivos se ha encontrado un aumento de cáncer de pecho y en el caso de zonas cercanas a industrias de fertilizantes un incremento de cáncer de pleura [32].

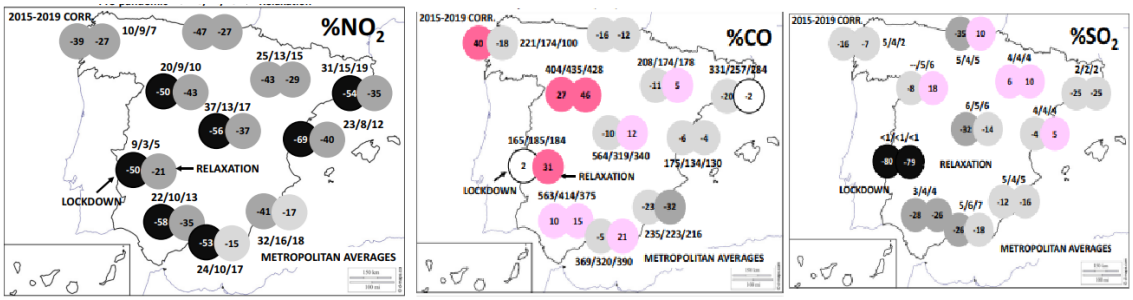


Figura 11: Cambio porcentual promedio de a) NO_2, b) CO y c) SO_2 respecto a 2015-2019; círculos indican concentraciones promedio ($\mu g/m^3$) durante prepandemia, confinamiento y relajación. Referencias:[33]

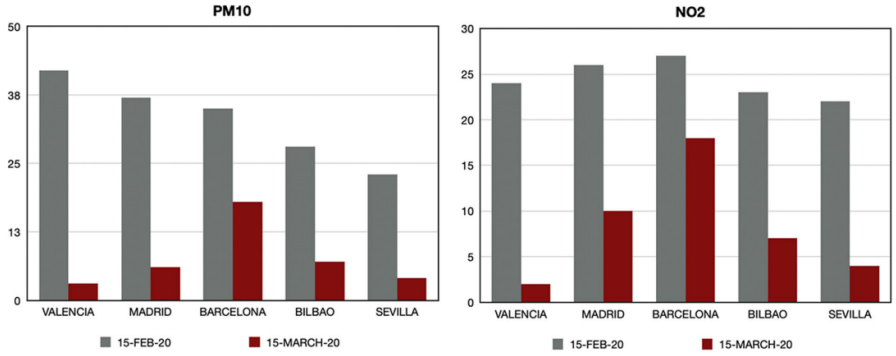


Figura 12: Concentraciones de PM_10 y NO_2 en principales ciudades españolas: gris antes de restricciones de movilidad por pandemia; rojo después del estado de alarma. Referencias:[34]

Las Fig.11 y Fig.12 permiten evaluar los cambios en las emisiones de NO_2, SO_2,

CO y PM_10 en períodos tan críticos como la pandemia. En las figuras se pueden observar cómo los porcentajes de emisiones de estos agentes contaminantes han disminuido drásticamente.

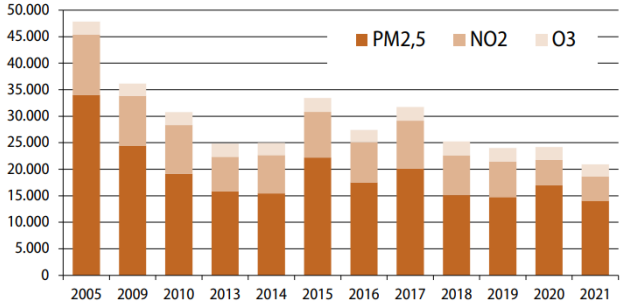


Figura 13: Muertes atribuibles a la contaminación en España. Referencias:[30]

En la Fig.13 se muestra una gráfica de la evolución de las muertes asociadas a la contaminación en España desde 2005 a 2021, en ésta se muestra una clara disminución de la mortalidad como respuesta a la disminución de las emisiones siguiendo una normativa cada vez más restrictiva, y, teniendo en cuenta también, la crisis del COVID-19 entre los años 2020 y 2021.

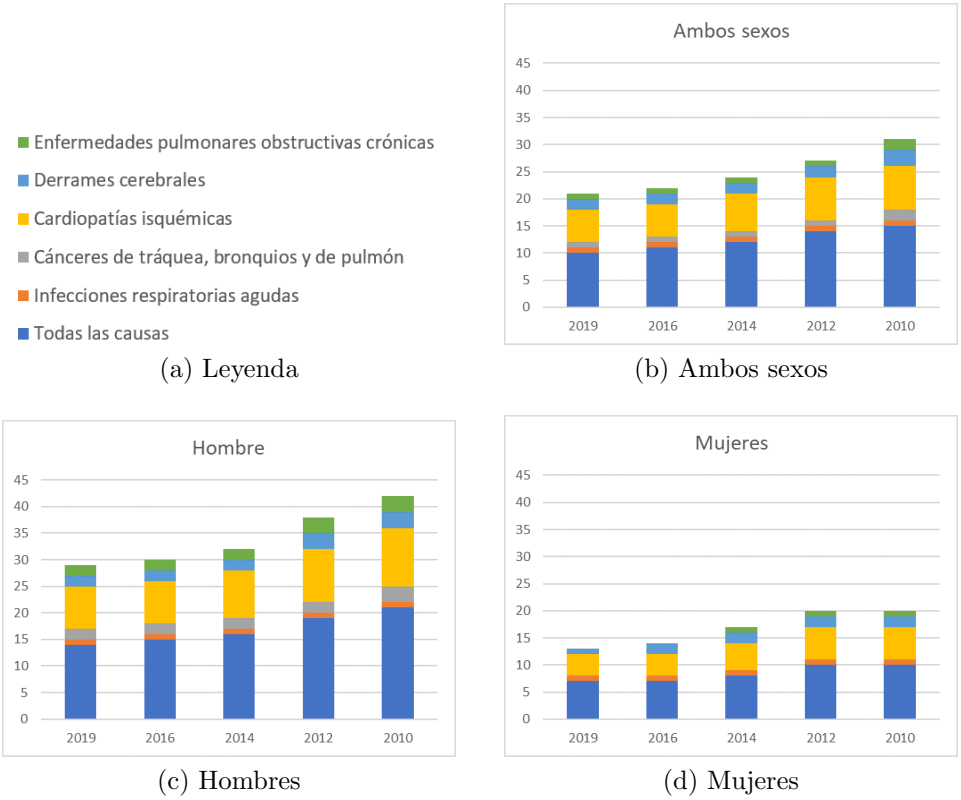
Observando todas las figuras expuestas anteriormente (Fig.11, 12 y13) se puede observar que hay una correlación directa entre las muertes atribuidas a los agentes contaminantes y la disminución de la presencia de éstos en los años de pandemia. Esto sugiere que las políticas y medidas extraordinarias adoptadas durante este período tuvieron un efecto positivo tanto en la calidad del aire como en la salud de la población.

En España se estimaron en 1996 unas 2696 muertes asociadas a problemas respiratorios y circulatorios que podrían estar relacionados con la contaminación ambiental, en Madrid. Con una media de edad de 77 años [35].

Según la Organización Mundial de la Salud (WHO), España cuenta con una de las menores concentraciones (medidas en $[\mu g/m^3]$) de partículas finas ($PM_{2.5}$) en 2019, con un promedio total de 9.34 $[\mu g/m^3]$. Al desglosar por áreas, las concentraciones eran de 9.77 en zonas urbanas, un 8,28 en zonas rurales y un 10.19 en ciudades. En comparación, en 2010 las concentraciones eran de 12,72 en promedio total, 13,35 en zonas urbanas, 11,20 en zonas rurales y 13,97 en ciudades [36].

En términos comparativos con otros países, mientras que España registraba una concentración total de 9,34 en 2019, Francia tenía 10,46, Alemania 10,73, Polonia

18,83, Italia 14,22 e Inglaterra 9,52 [36].



(a) Leyenda (b) Ambos sexos

(c) Hombres (d) Mujeres

Figura 14: Mortalidad en España asociada con la contaminación ambiental, ratio por cada 100.000 habitantes en los años 2010-2019. Referencias: elaboración propia

La Fig.14 se ha realizado teniendo en cuenta los datos obtenidos de [37], en esta se muestra el ratio de muertes por cada 100.000 habitantes asociadas con la contaminación ambiental en España. Al observar las gráficas se puede interpretar que la mortalidad en hombres es consistentemente mayor que en el caso de mujeres a lo largo de los años, lo que refleja una tendencia general en los datos. Además, se observa que la causa de mortalidad más frecuente relacionada con la contaminación en España las son cardiopatías isquémicas, que presentan una tasa de mortalidad considerablemente mayor en comparación con otras causas, como infecciones respiratorias agudas o enfermedades pulmonares obstructivas crónicas.

Los países que presentan una tasa de mortalidad más alta asociada a la contaminación de manera general son Tajikistan 160 muertes (ratio por cada 100.000 habitantes), Afganistán (145 muertes), Mongolia (120 muertes), Yemen (111 muertes) y países del África central (Chad, Camerún, Egipto, entre otros) [37].

2.2. Modelos de *Machine Learning* para la Predicción de Contaminación

Las técnicas de *Machine Learning* juegan un papel muy significativo en el entrenamiento de ordenadores o sistemas con el fin de predecir y tomar decisiones de una manera eficiente. El *Machine Learning* es el campo de experimentación científica y computacional que se dedica en el diseño de diferentes técnicas y aplicaciones que permiten aprender de una base de datos [38].

Este concepto, es decir, el campo del ML fue concebido al menos siete décadas atrás por parte de Arthur Samuel con el objetivo de desarrollar métodos computacionales que permitieran implementar diferentes formas de aprendizaje, en particular, mecanismos capaces de crear nuevo conocimiento a partir de datos de ejemplo. Uno de los inventos vitales de la investigación de la inteligencia artificial es la idea de que los problemas intratables se pueden resolver ampliando el esquema tradicional de

$$programa = algoritmo + datos \tag{1}$$

a uno más elaborado

$$programa = algoritmo + datos + reconocimientodelcampo \tag{2}$$

El funcionamiento de un algoritmo de *Machine Learning* se ve reflejado en la Fig.15. El sistema de aprendizaje determina una descripción de un concepto dado a partir de una serie de ejemplos y conocimientos previos [39].

El conocimiento previo contiene información sobre el lenguaje usado para describir los ejemplos y conceptos. El algoritmo de aprendizaje entonces se basa en el tipo de ejemplos, en el tamaño y la relevancia del conocimiento previo, sobre las cuestiones de representación, sobre la naturaleza presunta del concepto a adquirir, y sobre la experiencia del diseñador [39].

Los algoritmos de ML pueden ser de forma general clasificados en dos grupos [39]:

- **Métodos de caja negra:** como redes neuronales y estadísticas matemáticas. Este enfoque desarrolla su propia representación conceptual para propósitos de reconocimiento de conceptos. Sin embargo, esta descripción interna no puede ser fácilmente interpretada por el usuario y no proporciona ni información ni explicación sobre el proceso de reconocimiento. Los métodos de caja negra suelen involucrar cálculos numéricos de coeficientes, distancias o pesos.
- **Métodos orientados al conocimiento:** crean estructuras simbólicas de conocimiento que cumplan con el principio de comprensibilidad. Los sistemas de aprendizaje se dividen en tres criterios: débil, fuerte y ultra-fuerte.

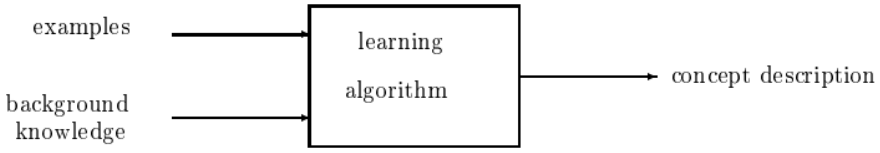


Figura 15: Funcionamiento de un algoritmo de *Machine Learning*. Referencias:[39]

Por otro lado, también se pueden usar modelos de *Deep Learning* los cuales se tratan de un tipo de *Machine Learning* que potencia los sistemas con el fin de conseguir mejores conclusiones e ideas. La Fig.16 muestra la relación entre estos conceptos y la Inteligencia Artificial [38].

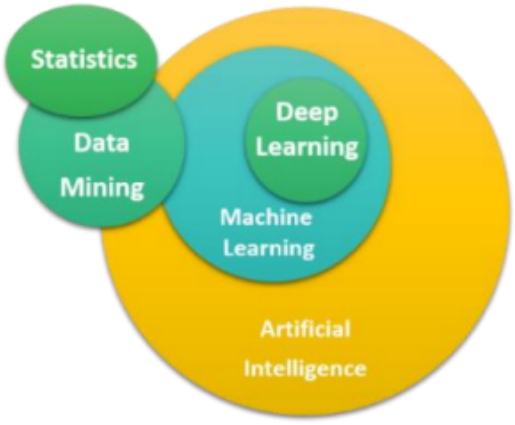


Figura 16: Relación entre *Machine Learning* y el resto de campos. Referencias:[40]

Los algoritmos de *Machine Learning* se dividen en dos grupos: supervisados y no supervisados. En el caso de algoritmos supervisados la base de datos cuenta, además

de con los datos con los que se realizarán las predicciones, con las conclusiones a las que debería llegar el algoritmo. Es decir, el aprendizaje supervisado guiará al algoritmo a encontrar la relación de asociación entre los datos y las conclusiones a las que debe llegar [10]. Ejemplos de algoritmos de aprendizaje supervisado son: *Random Forest (RF), Support Vector Machine (SVM) y Extreme Gradient Boosting (XGBoost)*

Por otro lado, el aprendizaje no supervisado no contará en la base de datos las conclusiones a las que debería llegar el algoritmo. Este tipo de *Machine Learning* es principalmente utilizado para explorar los datos, encontrar patrones entre ellos y clasificar. Un ejemplo de este tipo de algoritmo es *K-Means clustering* [10].

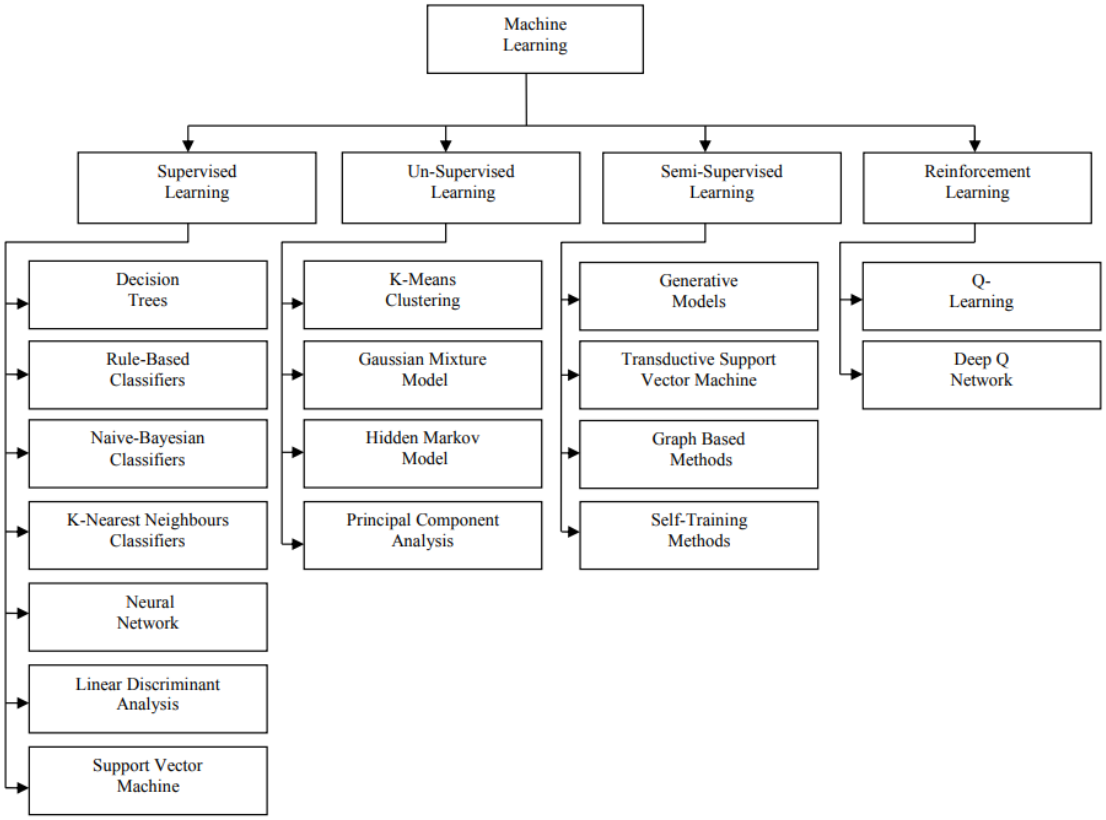


Figura 17: Clasificación de los diferentes algoritmos de *Machine Learning*. Referencias:[41]

En la Fig.17 se presenta una clasificación detallada y estructurada de los principales algoritmos de *Machine Learning*, los cuales se han desarrollado y refinado a lo largo del tiempo para abordar problemas en diversas áreas del conocimiento. Esta representación visual resulta útil no solo para entender las relaciones entre los

distintos métodos, sino también para elegir el algoritmo más adecuado dependiendo del tipo de problema, la naturaleza de los datos y los resultados esperados.

El proceso genérico de aplicación de algoritmos de *Machine Learning* consiste en siete pasos [40]:

- Recolección de datos: se trata de un paso importante porque determinará la capacidad de predicción del modelo.
- Limpieza de datos y pre-procesamiento: en muchas ocasiones los datos se encontrarán sin estructura, con mucho ruido o deben tomar otras formas para ser usados con algoritmos de ML. Por lo tanto, los datos necesitarán ser tratados con anterioridad.
- Ingeniería de características: este será el siguiente paso, en el que se escogerán las características más relevantes de la base de datos.
- Definición de modelo de *Machine Learning*: este proceso comprenderá la investigación y elección del mejor método de ML para los datos que se poseen.
- Entrenamiento: en este paso se utilizará una parte de los datos para mejorar la capacidad de predicción del algoritmo.
- Evaluación de funcionamiento: posterior al entrenamiento será necesario observar cómo funciona el algoritmo en la otra parte de los datos. En ocasiones es necesario volver un paso atrás para volver a entrenar el algoritmo.
- Predicción: este último paso se trata de observar los resultados obtenidos y obtener conclusiones.

Algunos de los principales algoritmos de ML son los siguientes [42]:

- **Algoritmo de Descenso de Gradiente (*Gradient Descent Algorithm (SGD)*):** método iterativo que tiene como objetivo minimizar el coste de la función. El principal beneficio del uso de este algoritmo es la eficiencia computacional y, como desventaja, en ocasiones el algoritmo puede converger en un punto que sea óptimo.

- **Regresión linear (*Linear Regression Algorithm*):** puede resultar útil para modelar variables continuas y obtener predicciones como, por ejemplo, la predicción de las notas de los alumnos o el pronóstico de las ventas. Los principales beneficios de este algoritmo será la comprensión simple y la simplicidad para evitar el *overfitting*. Respecto a las desventajas es que no realizará un buen ajuste cuando se produzcan relaciones no lineares entre variables.

- **Regresión multivariable (*Multivariate Regression Analysis*):** generalmente los problemas reales son demasiado complejos como para realizar una regresión linear simple ya que una sola variable puede depender de numerosos factores. En este tipo de algoritmos permitirá establecer más de una relación entre un número de variables independientes y una variable dependiente. Los beneficios que posee son la posibilidad de establecer relaciones entre variables independientes y modelar problemas más complejos y del día a día. Sin embargo, también cuenta con desventajas como lo son la complejidad de la técnica y que el número de muestras que necesita el modelo debe ser alto.

- **Árboles de decisión (*Decision tree*):** se trata un método de ML supervisado que permite resolver problemas de clasificación y regresión por división continua de datos en función de un parámetro determinado. En los árboles de decisión las variables son categóricas. Las ventajas que presenta este método son, la fácil interpretación de los resultados, la posibilidad de hacer uso de valores categóricos y cuantitativos y la eficiencia del método. Por otro lado, las desventajas son la dificultad a la hora de controlar el tamaño de los árboles y que las soluciones son resultado de puntos óptimos locales y no globales.

- **SVM (*Support Vector Machine*):** este método soporta también problemas de clasificación y regresión. Este tipo de método necesita de definir cuales va a ser los límites de decisión. Los beneficios son: el algoritmo soporta datos semi-estructurados, estructurados y funciones complejas, esto resulta en una menor probabilidad de producir *overfitting*. Por otro lado, las desventajas: su rendimiento disminuye con un exceso de datos y, además, no funciona bien cuando el conjunto de los datos tiene ruido.

- **NB (*Naïve Bayes):*** se trata de un algoritmo sencillo y basado en probabilidades condicionales. En este método se hará uso de una tabla de probabilidades y de la base de datos. Las ventajas de este método: implementación sencilla,

buen funcionamiento, funciona bien con una baja cantidad de datos, etc. Por otro lado, las desventajas: es difícil aplicar NB directamente si se necesita trabajar con variables continuas (como el tiempo), en comparación con otros algoritmos como SVM o regresión logística simple, Naïve Bayes requiere más memoria en tiempo de ejecución durante las predicciones.

- **K-Means:** frecuentemente utilizado para resolver problemas de clasificación. Se trata de un algoritmo de aprendizaje no supervisado. Las ventajas que presenta son: alta eficiencia computacional cuando el número de variables es grande y fácil implementación. Las desventajas: predecir el número de *clusters* puede resultar difícil y, además, el rendimiento del algoritmo disminuye cuando los *clusters* no son de forma esférica o globular.

2.2.1. Aplicaciones de los algoritmos de *Machine Learning*

La evolución del ML fue propulsada por Arthur Samuel en 1959 el cual introdujo el término de *Machine Learning*. Antes de esto, sus aplicaciones eran para juegos de ajedrez basados en programas de ordenador. Hetch mostró la idea de multicapa (MLP) con entrenamiento (BP). Y, avanzando a tiempos actuales, surge una nueva era de redes neuronales llamada *Deep Learning*. El tercer auge de estas redes neuronales comenzó en 2005 con investigadores como Andrew Ng, Hinton, Bengio, LeCun entre otros [43].

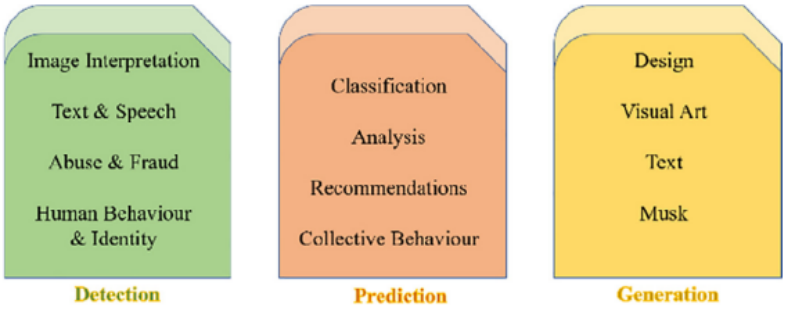


Figura 18: Aplicaciones de *Machine Learning* y *Deep Learning*. Referencias:[43]

En la Fig.18 se muestran los diferentes campos de aplicación del ML.

Sharma (2021) habla en su artículo sobre diferentes aplicaciones de las técnicas de ML como por ejemplo la visión por computadora (*Computer Vision*) la cual permite la detección de objetos, reconocimiento facial, usado por ejemplo para la crisis del

COVID-19 o para identificación de extremistas en lugares concurridos (aeropuertos, congresos, etc.) entre otras aplicaciones. Dentro de los tipos de reconocimientos también se hace uso de reconocimientos de escritura a mano (*Handwritten Recognition*) la cual es usada para digitalizar documentos manuscritos y, reconocimiento por voz (*Speech Recognition*) que permiten traducción de palabras a texto, atención médica, interfaces y asistentes de voz, etc.

Otras aplicaciones pueden ser la detección precisa de variaciones humanas de salud en tiempo real, procesamiento de parámetros médicos y registros y análisis estadístico de la documentación médica. O, predicciones basadas en datos históricos como el precio de las acciones, investigación científica y diagnósticos médicos [43].

En el campo de la salud las aplicaciones de algoritmos de ML son numerosas. Suja (2022) expone en su artículo [44] una revisión sobre el uso de técnicas de ML en la salud humana. Suja (2022) trae a colación la gran mejora que ha supuesto este tipo de mecanismos para los estudios en la salud humana como, por ejemplo, la posibilidad de realizar predicciones médicas a partir de datos no médicos (EHR), la adquisición de conocimiento relacionado con diagnósticos médicos o la identificación de características comunes en enfermedades.

3. Fundamentos teóricos

3.1. Aprendizaje supervisado

3.1.1. Regresión lineal

Uno de los métodos principales dentro del aprendizaje supervisado es la regresión lineal, la cual ofrece un enfoque estadístico en el cual se modela la relación entre una variable dependiente y una o más variables independientes. El objetivo, disminuir el error entre las predicciones del modelo y los valores reales observados. En la Fig.19 se muestra representado un ejemplo de regresión lineal simple.

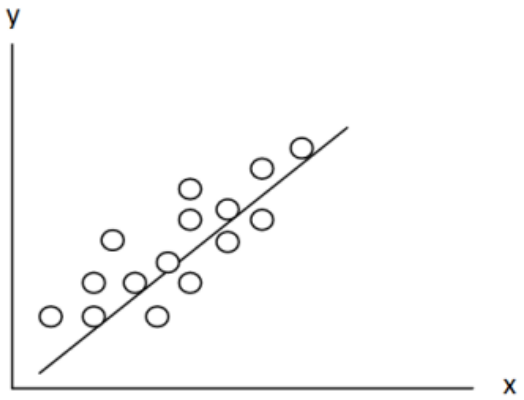


Figura 19: Regresión lineal simple. Referencias:[45]

El análisis por regresión lineal puede ser dividido en dos: regresión lineal simple y regresión lineal múltiple.

A. Regresión lineal simple

En el caso de la regresión lineal simple la relación entre la variable dependiente "yz la independiente "$x_i(i = 1, 2, 3...)$". El parámetro "a_0"se tratará de una constante que indica el punto de intersección de la línea de regresión con el eje vertical y "a_1"se trata de un coeficiente de regresión que indica la pendiente de la recta de regresión. Por último, "e"se trata de un error aleatorio, utilizado para expresar el efecto de ciertos factores aleatorios en la variable dependiente [46].

$$y = a_0 + a_1 \cdot x + e \tag{3}$$

Las distancias entre los puntos y la línea de regresión será "e", es decir, el error o residuo. Entre más cerca se encuentren los puntos a la línea, mejor será el ajuste entre la línea de regresión y el dato. Es decir, este residuo permitirá verificar la ecuación con el fin de comprobar cuan bien se ajusta la línea a los datos.

En la regresión se utilizan los llamados mínimos cuadrados, también conocidos como mínimos cuadrados de regresión, los cuales determinan la línea que minimiza la suma de las distancias verticales cuadradas, desde los puntos hasta la recta [47]. En la Fig.20 se muestra gráficamente este concepto.

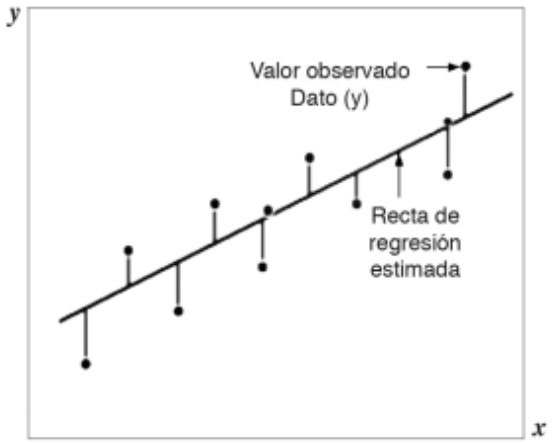


Figura 20: Representación de los mínimos cuadrados. Referencias:[47]

El análisis por regresión evaluará "a_0z "a_1.ºbservando los datos aportados "(x_i, y_i)". Este método dibujará diagramas de dispersión entre la variable dependiente e independiente. Después de esto, la ecuación de regresión y la precisión estará definida por la bondad del ajuste o "*Goodness of fit*".

B. Regresión lineal múltiple

La regresión lineal simple tendrá una variable dependiente guiada por una sola variable independiente. Sin embargo, generalmente los problemas tendrán más de una variable independiente que afecta a la dependiente o de salida "y". Por ejemplo, el precio de una casa depende de numerosos factores como el vecindario, el área que posee, el número de habitaciones, etc. En resumen, en la regresión lineal simple habrá una sola relación entre la variable de entrada y la de salida. Mientras que, en la regresión lineal múltiple existirá más de una relación entre el número de variables independientes (entradas/predictivas) y una variable dependiente (salida/respuesta) [42].

La presencia de un mayor número de variables independientes no indica que la regresión será mejor. La regresión simple y múltiple tendrán diferentes usos y una no será superior a la otra. En algunos casos incluir más de una variable de entrada podrá empeorar la situación y resultar en "*overfitting*". En un escenario óptimo todas las variables de entrada tendrán relación con la variable de salida, pero no entre ellas [42].

Es decir, el modelo de regresión lineal múltiple se utilizará cuando:

- La variable dependiente "y"depende linealmente de cada una de las variables explicativas o de entrada "$(x_1, x_2, ..., x_n)$".
- Una variable no basta para explicar con exactitud la variable "y".

La regresión lineal multi-variable se encuentra representada en la Fig.21. Ésta se trata de un proceso de predicción con más de una variable independiente o variables predictivas.

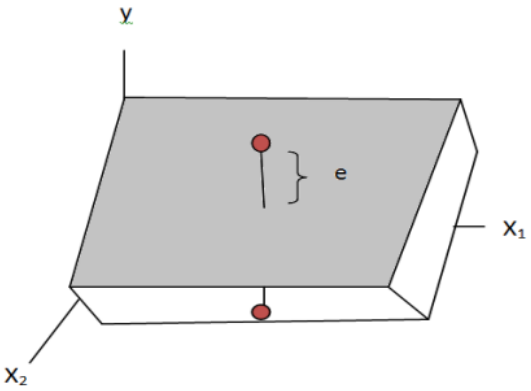


Figura 21: Regresión lineal multivariable. Referencias:[45]

$$y = a_0 + a_1 \cdot x + a_2 \cdot x + ... + a_n \cdot x + e \tag{4}$$

En el caso particular de que haya dos variables independiente el modelo tendrá la forma mostrada en la ecuación 5.

$$y = a_0 + a_1 x_1 + a_2 x_2 + e \tag{5}$$

Gráficamente, el modelo de regresión lineal con dos variables independientes supone calcular la ecuación de un plano que describe la relación de "$y$çon "x_1ז "x_2". En la Fig.22 se muestra la relación de "$y$çon "x_1ז "x_2".

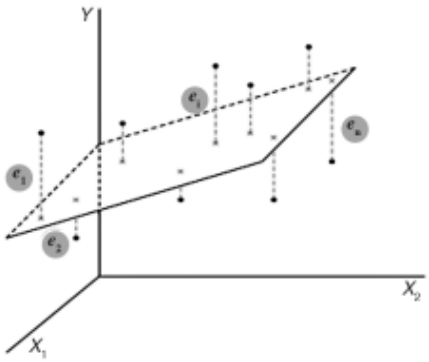


Figura 22: Modelo de regresión lineal con dos variables independientes, mínimos cuadrados. Referencias:[47]

C. Regresión polinómica

La regresión polinómica se trata de un caso especial de regresión múltiple, con únicamente una variable independiente [48]. La regresión polinómica con una variable se podrá expresar como se muestra en la ecuación 6.

$$y_i = a_0 + a_1 x_i + a_2 x_i^2 + a_3 x_i^3 + ... + a_n x_n^k + e_i \tag{6}$$

para i = 1,2,...n, y donde "k"se trata del grado del polinomio o el orden del modelo. Este tipo de regresión se trata, como ya se ha comentado, de un caso similar a la regresión lineal múltiple pero con $x_1 = x, x_2 = x^2, x_3 = x^3$, etc.

Los modelos de regresión polinominal se usan cuando la variable de respuesta muestra un comportamiento curvilíneo o no lineal [49].

D. Formulación matemática: regresión y álgebra lineal

Para la obtención de la regresión lineal múltiple, es decir, con diversas variables independientes, es posible plantear un modelo matricial tal y como se muestra en la ecuación 7.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix} \tag{7}$$

De este modo, mediante la notación de matrices se podría expresar la ecuación 7, tal y como se muestra en la ecuación 8.

$$Y = X \cdot a + e \tag{8}$$

Dado que el fin de la ecuación es encontrar el vector de coeficientes "a", es posible mediante las leyes de las matrices expresar la ecuación 8 en términos de la ecuación 9.

$$a = (X^T X)^{-1} X^T Y \tag{9}$$

Así, el vector resultante "$a$çontiene los diferentes coeficientes $(a_0, a_1, a_2, ..., a_p)$ [50].

E. Goodness of fit

En modelos de regresión, la bondad de ajuste se refiere a la capacidad del modelo para describir adecuadamente la relación entre las variables. Para evaluar qué tan bien el modelo predice los valores observados, se utilizan diversas métricas, siendo una de las más comunes el coeficiente de determinación "R^2". Este coeficiente permite medir la proporción de la variabilidad de la variable dependiente que es explicada por la variable independiente, proporcionando una indicación de la efectividad del modelo.

$$R^2 = \text{cor}^2(\hat{Y}, Y) \tag{10}$$

$$R^2 = \frac{ESS}{TSS} = 1 - \frac{ESS}{TSS} \tag{11}$$

$$TSS = \sum (Y_i - \bar{Y})^2 \tag{12}$$

$$TSS = \sum (\bar{Y}_i - \bar{Y})^2 \tag{13}$$

$$TSS = \sum (Y_i - \bar{Y}_i)^2 \tag{14}$$

Para las ecuaciones anteriores $\hat{y}$ será el valor original e y será el valor previsto, por lo que la ecuación TSS será la suma de cuadrados para el total y ESS será la suma de cuadrados de la regresión. Además, RSS será la suma residual de cuadrados [46].

El coeficiente "R^2"se puede interpretar como el ratio de varianza de la variable dependiente predecible en la variable independiente. Este coeficiente pordrá tomar valores entre $-\infty$ and 1 según la relación entre la realidad y lo predicho por el modelo. Se suele asumir el valor de "R^2çomo un valor mayor o igual a cero, ya que, cuando es negativo, éste indica que el modelo de regresión tiene un rendimiento peor que simplemente el us de la media de los datos como predicción [51]. Es decir:

- R^2 **positivo o cercano a 1:** el modelo explica bien la variabilidad de los datos.
- R^2 **es igual a 0:** el modelo no tiene capacidad predictiva y simplemente usar la media de los valores de salida "y"daría el mismo resultado.
- R^2 **es negativo:** indica que el modelo tiene peor desempeño que la simple predicción de los datos utilizando el promedio de éstos.

3.1.2. Redes neuronales

El cerebro está compuesto de redes de neuronas, éstas, reciben entradas de otras neuronas en forma de excitación, cuando esta excitación alcanza un umbral, la neurona se dispara y el proceso se repite. Como tal, la salida de una neurona siempre lleva la misma relación que su entrada [52].

En 1940, Warren McCulloch y Walter Pitts exploraron las habilidades computacionales de las redes neuronales (Fig.23). Combinando esas neuronas sería posible

construir redes que procesen funciones Booleanas sencillas (ej, "Soy un hombre, y los hombres son mortales. Por tanto, soy mortal.").

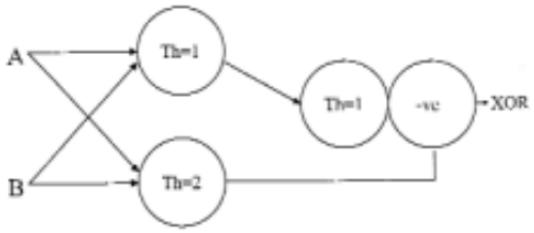


Figura 23: Redes neuronales de McCulloch-Pitts. Referencias:[52]

El siguiente paso importante en el desarrollo de las redes neuronales fue realizado por Rosenblatt, que, como psicólogo, vio el cerebro como un asociador del aprendizaje que clasifica estímulos y respuesta a estos. En lugar de considerar la cognición como una función de lógica simbólica, Rosenblatt propuso un enfoque más flexible basado en la separabilidad estadística [52].

Para lograr esto, desarrolló una serie de redes denominadas "perceptrones", los cuales constan de tres capas de neuronas [52]. Mostradas en la Fig.24.

- **Capa de entrada:** será la encargada de recibir los estímulos.
- **Capa de asociación:** donde las conexiones entre neuronas son parciales y aleatorias.
- **Capa de salida:** donde las neuronas generan la respuesta final de la red.

Estas capas de neuronas se encontrarán conectadas por conexiones aleatorias, permitiendo que la red aprenda patrones y clasifique datos en función de los estímulos recibidos [52].

Cada nodo, es decir, cada neurona tendrá una estructura interna como la mostrada en la Fig.25. Esta figura muestra que la entrada a la neurona (x_i) y los pesos asociados (w_i), es decir, coeficientes que modulan la influencia de de cada entrada en el resultado final de la neurona (Y) [53].

El bloque fundamental de una red neuronal es una única capa computacional. Es decir, una unidad recibirá un grupo de números reales como entrada, los tratará y producirá una única salida [54].

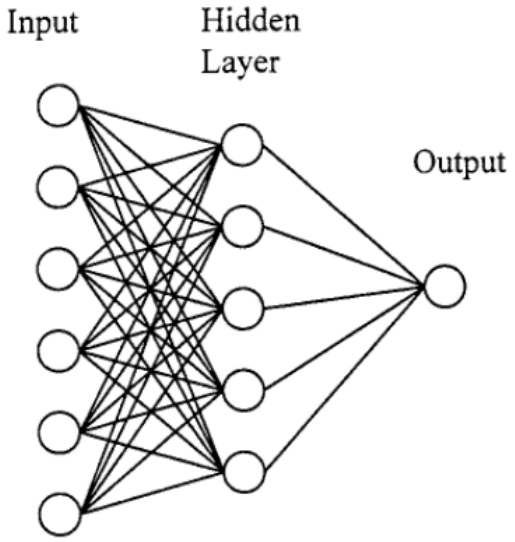


Figura 24: Representación gráfica de redes neuronales. Referencias:[52]

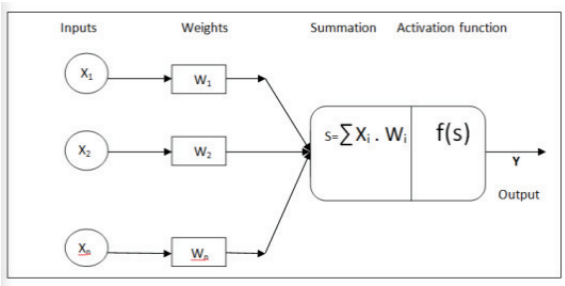


Figura 25: La estructura de una neurona. Referencias:[53]

En esencia, una unidad neuronal tomará una suma ponderada de sus entradas con un término adicional en la suma denominado *bias term*. Las entradas se denominarán $x_1, x_2, ...x_n$, una unidad tiene un conjunto correspondiente de pesos $\omega_1, \omega_2, ..., \omega_n$ y un *bias* b, por lo que la suma ponderada z se podrá representar como [54]:

$$z = b + \sum_i \omega_i x_i \tag{15}$$

Finalmente, en lugar de hacer uso de z, y por una función lineal x como salida, las redes neuronales aplicarán una función no lineal f a z. Nos referimos a la salida de esta función no lineal como valor de activación de la unidad a. En el caso de que se modele una sola unidad, la activación del nodo es, de hecho, la salida final de la red, a la que generalmente denominaremos y (Fig.25). Este valor se definirá por

tanto como [54]:

$$y = a = f(z) \tag{16}$$

Hay tres tipos de funciones no lineales comunes, la sigmoide, la tan h y la unidad lineal rectificada (ReLU). Entre éstas, la más común para comenzar será la función sigmoide:

$$y = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{17}$$

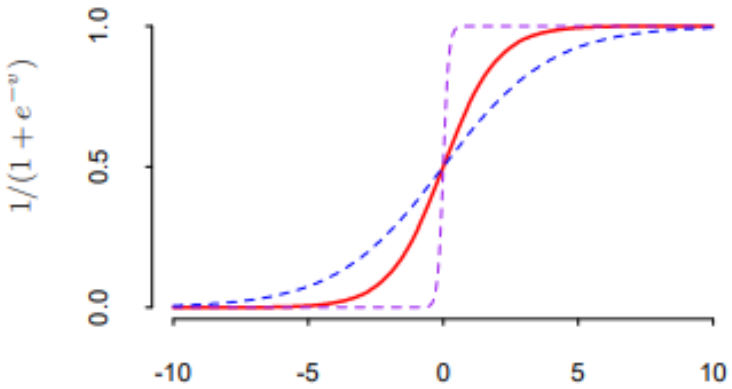


Figura 26: Gráfica de representación de la función sigmoide. Referencias:[55]

Por otro lado la función tahn, mostrada en la Fig.27a se trata de una variante de la función sigmoide en los rango entre -1 hasta +1:

$$y = tahn(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{18}$$

Y la función de activación más simple, ReLU, mostrada en la Fig.27b será la misma que z cuando z sera un número positivo o 0:

$$y = ReLU(z) = max(z, 0) \tag{19}$$

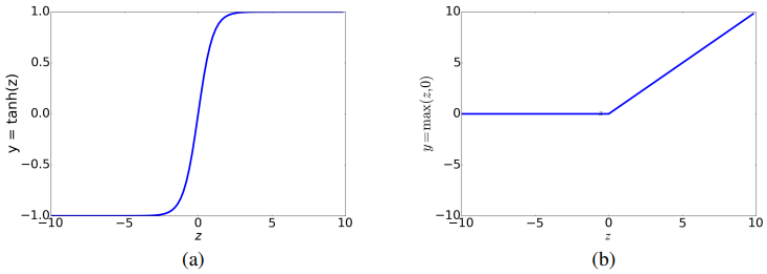


Figura 27: Gráfica de representación de las funciones a) tahn y b) ReLU. Referencias:[54]

A. Redes Neuronales predictivas

Como se ha comentado anteriormente existen también redes neuronales que poseen múltiples capas (Fig.24). Este tipo de redes neuronales se denominan redes neuronales predictivas (*feedforward network*), ésta se trata de la forma más simple de red neuronal, en ella las unidades se conectarán sin círculos, es decir, la salida de una unidad pasará a unidades de la siguiente red y no volverán a la red anterior en ningún caso. También se denomina en otras ocasiones como perceptrones multicapa (*MLP*) [54].

Las salidas de los núcleos de la capa oculta formarán un vector denominado h, por otro lado también se encontrará el vector de *bias* b y el vector de entrada x. Por otro lado la matriz de peso W tendrá tantas columnas como entradas (x_i) y tantas filas como *bias* (b_i) y representará los pesos que conectan la capa de entrada con la capa oculta. Aplicando la función sigmoide la ecuación queda de la siguiente manera:

$$h = \sigma(Wx + b) \tag{20}$$

También se deberá calcular la suma ponderada de las activaciones de la capa oculta usando otro tipo de pesos U con el fin de obtener un valor intermedio z. El valor h será el vector de activaciones de la capa oculta obtenido en el paso anterior.

$$z = Uh \tag{21}$$

Finalmente, será necesario normalizar el vector z con el fin de convertirlo en una distribución de probabilidades, que será la salida final de la red (y). Ésta salida tendrá información sobre las probabilidades de cada clase. Se hará uso de la función $softmax$ que asegurará que todos los valores de y se encuentren entre 0 y 1 y, además, sumen 1, lo que hace que este método también sea útil para problemas de clasificación.

$$y = softmax(z) \tag{22}$$

En la Fig.28 se muestran gráficamente las diferentes variables implicadas en las ecuaciones planteadas anteriormente.

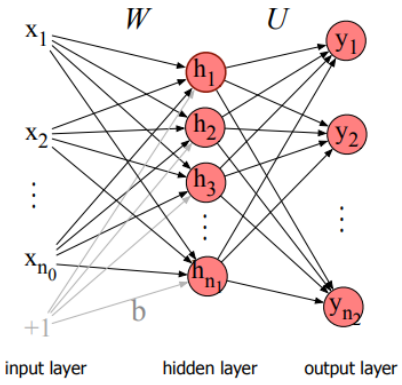


Figura 28: Variables de una red neuronal predictiva o *feedforward network*. Referencias:[54]

La necesidad de hacer uso de funciones de activación no lineales en cada una de las capas de una red neuronal proviene de que, en caso de no hacerlo, la red neuronal sería matemáticamente equivalente a una red de una sola capa, perdiendo así todo el poder representacional de tener múltiples capas [54].

B. Redes Neuronales Recurrentes

Una red neuronal recurrente (RNN) se trata de cualquier tipo de red neuronal que contiene un círculo en las conexiones de las redes neuronales. Esto significa que

el valor de alguna unidad depende directamente o indirectamente de sus propias salidas anteriores como entrada.

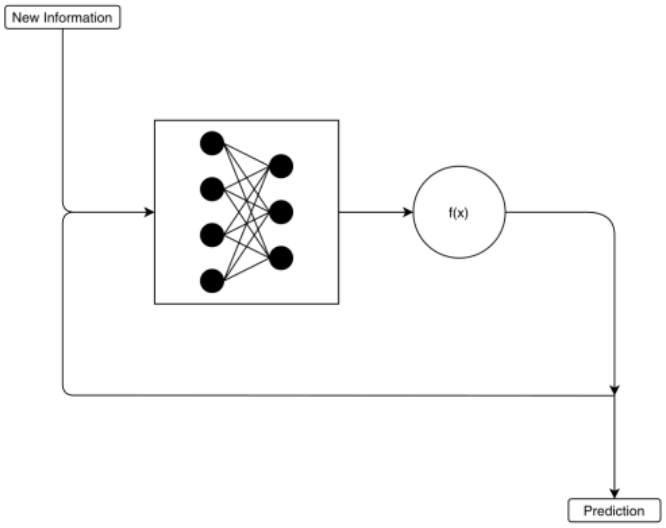


Figura 29: Diagrama de grafo cíclico de una Red Neuronal Recurrente. Referencias:[56]

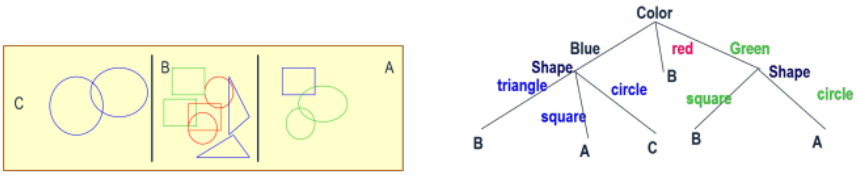
Las RNN poseerán un atributo relevante, comparten parámetros. Esto quiere decir que, en lugar de asignar parámetros distintos para cada dato de una secuencia (como haría una red neuronal tradicional), las RNN hará uso de los mismos parámetros para todos los datos de la secuencia. Esto permite procesar secuencias de longitud variable [56].

3.1.3. Árboles de decisión

Se trata de un algoritmo muy común para tomar diferentes decisiones por parte de los humanos. Se trata de un modelo simple de clasificación supervisada. Utilizada generalmente para clasificar las características de un objetivo único [57]

Los árboles de decisión son aquellos en los que cada rama (proveniente desde la raíz) representa una secuencia de datos que se divide hasta que se alcanza un resultado booleano en la hoja nodo [58]. Los árboles de decisión se trata de una estructura de datos jerárquica que representa datos a través de una estrategia de *divide y vencerás* [59]. Un ejemplo básico se muestra en la Fig.30.

La estructura de los árboles de decisión se basa en un grafo dirigido no cíclico con las siguientes partes:



(a) Example Data (b) Decision Tree

Figura 30: Ejemplo de árboles de decisión. Referencias:[59]

- **Nodos internos:** representan decisiones o pruebas basadas en una característica del conjunto de datos.
- **Ramas:** representan los resultados de las pruebas.
- **Hojas:** representan las predicciones finales del modelo.
- **Raíz:** se trata de la primera decisión a tomar donde el árbol se puede ramificar hasta alcanzar las hojas.

Considerando un problema de regresión donde la meta es predecir una variable "t"desde un vector de variables de entrada *D-dimensional* $x = (x_1, ..., x_D)^T$. Los datos de entrenamiento consistirá un vector de entradas $(x_1, ..., x_N)$ junto con las correspondientes etiquetas de atributos $(t_1, ..., t_N)$. Si se da la partición del espacio de entrada y buscamos minimizar la función de error, entonces el valor óptimo del predictivo variable dentro de cualquier región dada es dado por la media de los valores de t_n [60].

La dificultad de los árboles de decisión es saber cuando detenerse (no agregar más datos). Teóricamente la estrategia debe ser detener el árbol cuando la reducción del error residual sea pequeña. Pero esto no siempre es efectivo. Dado este problema una técnica es dejar crecer el árbol y luego "podarlo", es decir, eliminar aquellos nodos que no contribuyen a la mejora del modelo [60].

La predicción óptima para una región R_T será por tanto el valor promedio de los valores de t_n (valores reales de los datos). Se calcula como en la Eq.23.

$$y_\tau = \frac{1}{N_\tau} \sum_{x_n \in R_\tau} t_n \tag{23}$$

La contribución al error residual para cada nodo hoja se mide mediante la suma de los cuadrados residuales:

$$Q_\tau(T) = \sum_{x_n \in R_\tau} (t_n - y_\tau)^2 \tag{24}$$

El criterio de poda final, que busca minimizar el error residual total, será el siguiente:

$$C(T) = \sum_{\tau=1}^{|T|} Q_\tau(T) + \lambda |T| \tag{25}$$

Donde $Q_T(T)$ es el error residual de la región R_T asociada al nodo τ. Por otro lado λ se trata de un parámetro de regularización que penaliza la complejidad del modelo (número de nodos del árbol) y, por último $|T|$ el número total de nodos hoja del árbol.

El parámetro que controla el compromiso entre el error total (residual) y la complejidad del modelo se ajusta haciendo uso de la validación cruzada (Eq.26) o el índice de gini (Eq.27) [60].

La entropía cruzada mide la incertidumbre o desorden de una distribución de clases. Los parámetros $p_{\tau k}$ representan la proporción de puntos de datos en una región R_τ asignados a la clase k. Por otro lado, K se trata del número total de clases [60].

$$Q_\tau(T) = -\sum_{k=1}^{K} p_{\tau k} ln(p_{\tau k}) \tag{26}$$

El índice de Gini se trata de otra medida de impureza en la clasificación. Se hace uso de este método para medir cuán de homogéneo es el conjunto de datos dentro de una región R_τ [60]. Su fórmula es:

$$Q_\tau(T) = \sum_{k=1}^{K} p_{\tau k}(1 - p_{\tau k}) \tag{27}$$

En resumen la entropía cruzada se enfoca en medir la incertidumbre de las clases, y es más sensible a las distribuciones desbalanceadas de clases mientras que, el índice de Gini es computacionalmente más simple y suele ser el elegido cuando el objetivo es encontrar la mejor separación entre clases [60].

3.2. Aprendizaje no supervisado

3.2.1. K-Means

Como bien es sabido el *clustering* o clasificación es uno de las herramientas más importantes de la ciencia de datos. Éste es usado en diferentes campos como la bioinformática, departamentos de marketing, astronomía. etc. La meta de éste es dividir los datos en clases o *clusters* y agruparlos acorde a la distancia Euclidiana [61].

El agrupamiento por el algoritmo K-Means tiene también algunos defectos: en primer lugar, si el número de *clusters* que se selecciona nos es adecuado es fácil que el algoritmo caiga en un mínimo local. En segundo lugar, el algoritmo también es susceptible a valores atípicos que se usarán como centroides iniciales y su eficacia se verá significativamente reducida. Por último, los resultados de la agrupación se ven fácilmente afectados por el ruido y los valores atípicos lo cual hace que los resultados de la agrupación no sean adecuados [61].

El algoritmo de K-Means sigue un proceso muy sencillo, tal y como se muestra en la Fig.31:

- Seleccionar el número de puntos (K) como centroides iniciales.
- Repetir.
- Formar "K"grupos asignando cada punto a su centroide más cercano.
- Re-calcula el centroide de cada grupo hasta que los centroides no cambien.

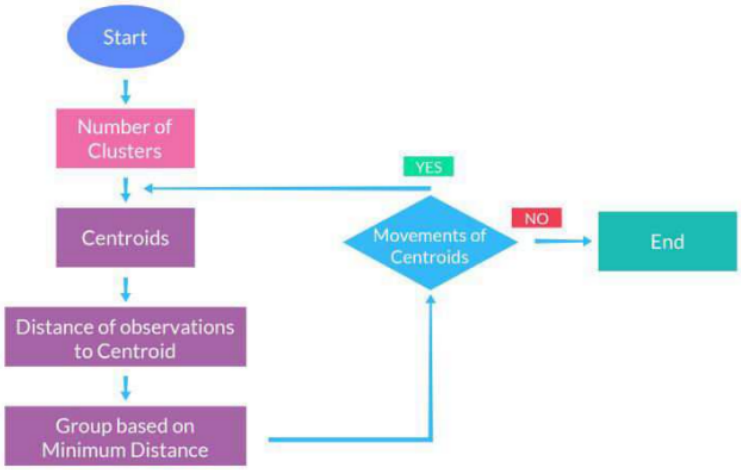


Figura 31: Algoritmo genérico de K-Means. Referencias:[62]

El algoritmo de K-Means toma la distancia como una unidad de medida estándar, genera K grupos en el conjunto de los datos, calcula el valor medio de la distancia y luego determina el centroide inicial. Cada grupo de define según su centroide. El objetivo será formar grupos a partir de n puntos de datos $(x_1, x_2, ..., x_n)$ en k conjuntos $(S_1, S_2, ..., S_k$ donde $k < n$ para minimizar el valor medio total (incluyendo la distancia cuadrada desde cada punto hasta su centroide) [61]. Por tanto, el objetivo de la optimización es encontrar:

$$argmin_S \frac{1}{n} \sum_{i=n}^{k} \sum_{x \in S_i} ||x - \mu_i||^2 \tag{28}$$

Donde μ_i es la media de los puntos en S_i.

El paso dos será asignar cada punto al grupo cuyo centroide sea el más cercano al punto, es decir, se debe satisfacer la siguiente fórmula:

$$S_i^{(t)} = \{x \mid ||x - \mu_i^{(t)}||^2 \leq ||x - \mu_j^{(t)}||^2, \ \forall j, \ 1 \leq j \leq k\} \tag{29}$$

$S_i \cap S_j = \emptyset, \quad \forall i, j \leq k$. En otras palabras, si un punto es equidistante a múltiples centroides, solo podrá ser asignado a uno de ellos.

El tercer paso será calcular el valor medio de todos los objetos en cada categoría se utiliza como el centroide del grupo para actualizar el centroide.

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i} x_j \tag{30}$$

El último paso será comprobar si los valores del centroide del grupo y la función objetivo han cambiado $\mu^{(t+1)} = \mu^{(t)} \quad \forall i \leq k$. Esto significa que, si se alcanza el número máximo de iteraciones, la asignación de grupos no cambiará en la siguiente actualización. De lo contrario, se debe volver al segundo paso [61].

3.2.2. Análisis de los principales componentes (PCA)

3.3. Métodos de interpolación en el mapa según el número de datos para el cálculo

Casi todos los métodos de interpolación se basan, en mayor o menor medida, en el uso de técnicas estadísticas. Por tanto, será necesario partir de unos conocimientos básicos de esta ciencia. Existen diferentes tipos de estadísticos para caracterizar una variable:

- Estadísticos de tendencia central: media, mediana y moda.
- Estadísticos de dispersión: rango, varianza y desviación típica.
- Estadísticos de forma: sesgo y curtosis.

De los más usados son:

- Media:

$$m_x = \frac{\sum_{i=1}^{N} X_i}{N} \tag{31}$$

- Varianza:

$$s_x^2 = \frac{\sum_{i=1}^{N} (X_i - m_x)^2}{N} \tag{32}$$

- Desviación típica

$$s_x = \sqrt{s_x^2} \tag{33}$$

- Media ponderada: donde "$W_i$representa el conjunto de coeficientes de ponderación cuya suma es 1. Esta media tiene como finalidad dar mayor importancia a alguno de los individuos.

$$m_x = \sum_{i=1}^{N} W_i * X_i \tag{34}$$

En el caos de que se quiera caracterizar el comportamiento conjunto de dos o más variables (X,Y,...). Los dos estadísticos fundamentales para esto casos son:

- Covarianza:

$$COV_{x,y} = \frac{\sum_{i=1}^{N}(X_i - m_x) * (Y_i - m_y)}{N} \tag{35}$$

- Coeficiente de correlación de Pearson:

$$r_{x,y} = \frac{COV_{x,y}}{s_x * s_y} \tag{36}$$

El *Coeficiente de Correlación* se introduce para obtener un valor estadístico que indique la dependencia entre dos variables. Los valores se encuentran comprendidos entre 1 (dos variables crecen al unísono) y -1 (cuando una variable crece la segunda decrece). En el caso de que la correlación sea 0 indica la ausencia de relación entre las variables.

En el caso de que se demuestre que dos variables tienen una relación (positiva o negativa) entre ellas, el siguiente paso será calcular los parámetros de una ecuación lineal de la forma:

$$Y = AX + B \tag{37}$$

esto permitirá estimar una de las variables (Y) a partir de la otra (X). Estos

coeficientes se calculan mediante las siguientes ecuaciones:

$$A = \frac{COV_{x,y}}{s_x^2} \tag{38}$$

$$B = m_y - A * m_x \tag{39}$$

En el caso del espacio, éste es bidimensional y no hay ninguna dirección preferente. Esto implica que los valores de auto correlación no tienen por que ser los mismos en todas las direcciones.

Una forma de estudiar la variabilidad espacial de una variable medida en un conjunto de puntos es el *semivariograma*. Para su cálculo se hace uso de los siguientes pasos:

- Determinar los pares de puntos posibles (en total $\sum_{i=1}^{n-1} i$), donde n es el tamaño de la muestra.
- Para cada par (i, j) anotar la distancia $d_{i,j}$ entre los puntos y el cuadrado de la diferencia de los valores $dZ^2 = (Z_i - Z_j)^2$.
- Cálculo para cada valor de h de la función:

$$\gamma(h) = \frac{\sum_{k=1}^{n_h} dZ_k^2}{2n_h} \tag{40}$$

  donde k hace referencia a cada uno de los pares.

Calculando $\gamma(h)$ para diferentes valores de h, se obtendrá un *semivariograma experimental*. La representación gráfica (h en abscisas y $\gamma(h)$ en ordenadas) proporciona un resumen de la estructura de variación de la variable. En la Fig.32 se muestra un ejemplo de representación gráfica de éste.

Los distintos métodos de interpolación se pueden dividir en dos tipos fundamentales:

- *Métodos globales:* utilizan toda la muestra para estimar el valor en cada nuevo punto.
- *Métodos locales:* utilizan solo los puntos de muestreo más cercanos.

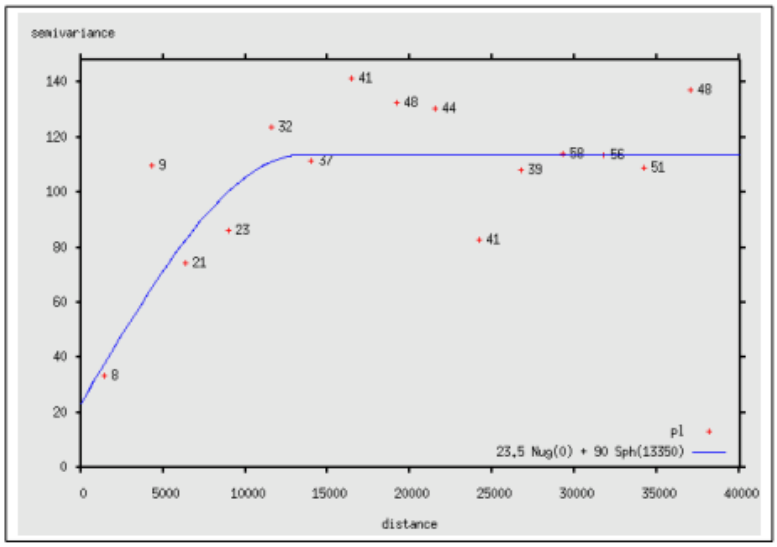


Figura 32: Representación gráfica de la función semivariograma. Referencias:[63]

3.3.1. Métodos globales

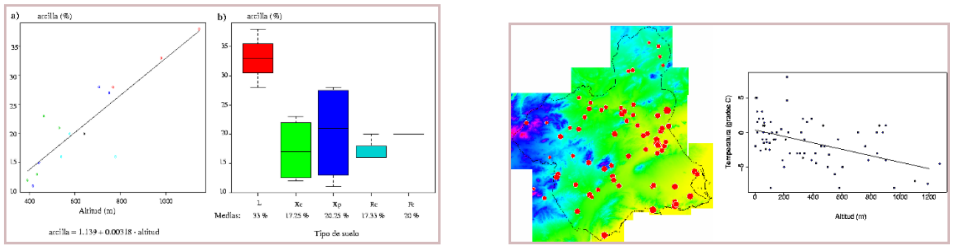
Asumen que la dependencia de la variable a interpolar de otras variables. En función del tipo de variables que se utilicen para determinar la variable a interpolar se darán dos casos distintos [63]:

Métodos de clasificación.

Para el caso de variables de apoyo cualitativas. En este caso en concreto la variable a interpolar adoptará en cada punto el valor medio correspondiente al valor de la variable de apoyo en ese punto [63].

Métodos de regresión. En este método se generará un modelo de interpolación de tipo polinómico. De modo general se hará uso de X e Y (longitud y latitud) como variables de apoyo y, además, otra variable cuantitativa "V"distribuida en el espacio. No se recomienda hacer uso de polinomios mayores a 3 ya que se podría producir sobreajuste [63]. En las Fig.33 se muestran gráficamente estos procedimientos.

Los métodos globales presentan un problema, y es que sólo consiguen capturar patrones generales de los datos y no tanto variaciones más pequeñas o locales. A modo de ejemplo, un método global podría identificar una tendencia general de que la temperatura es más alta en verano y más baja en invierno pero no capturaría detalles como que ciertas calles son más frescas por la sombra de los árboles. Por ello se podría aplicar un método global para eliminar la estructura general de los



(a) Modelo de regresión y clasificación del contenido en arcilla.

(b) Modelo de interpolación por regresión de la temperatura respecto a la altitud

Figura 33: Modelos de regresión. Referencias: [63]

datos y posteriormente un método local para observar las diferencias más específicas en cada punto [63].

3.3.2. Métodos locales

Los métodos locales están basados en el uso de los puntos más cercanos al punto específico de interpolación con el fin de estimar la variable "Z". Se denominará el conjunto de los datos cercanos *conjunto de interpolación*. Se asumirá una auto correlación espacial y se estimarán los valores de "Zçomo una media ponderada de los valores de un conjunto de puntos de muestreo cercanos.

El proceso será:

- Escoger aquellos puntos que van a formar parte del conjunto de interpolación, para esto se hará uso del criterio mostrado en la Fig.34. Se podrá hacer uso de dos métodos diferentes:
  - Escoger aquellos puntos cuya distancia al punto de interpolación sea inferior a un umbral r".
  - Escoger el número de puntos "n"que formarán parte del conjunto y seleccionar aquellos más cercanos al punto a interpolar.

  Una forma de obtener una distancia r.ªdecuada es haciendo uso de un *semivariograma*.
- Selección del método de interpolación.

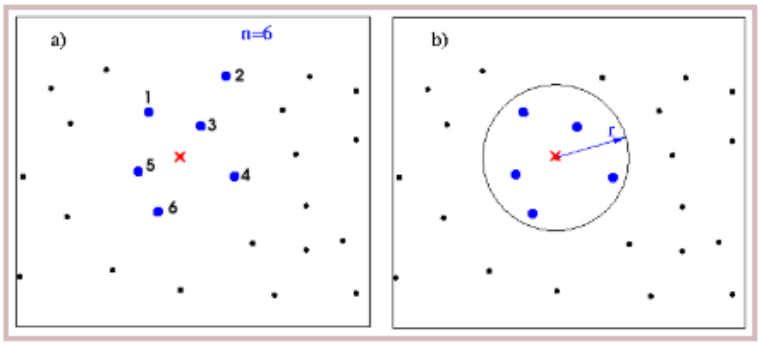


Figura 34: Criterios para obtener un conjunto de puntos de interpolación. Referencias:[63]

3.3.3. Determinación de las distancias

Todos los métodos de interpolación hacen uso de la medida de la distancia con el fin de determinar cómo de próximas están las muestras [64]. Se hace uso de dos tipos de distancias:

- **Distancia de Mahalonobis:** Tiene en cuenta la correlación entre las variables al utilizar la matriz de covarianza S. Si las variables están interrelacionadas, ajusta la medida de distancia en función de esa relación. Además, escala las diferencias en función de la varianza de cada variable.

$$d(x_1, x_2) = \sqrt{(x_1 - x_2)^T S^{-1} (x_1 - x_2)} \tag{41}$$

  donde S^1 es la inversa de la matriz de la covarianza de los datos.

- **Distancia Euclídea:** No considera la correlación entre las variables. Trata todas las dimensiones de la misma manera, sin importar la escala o la varianza de las características.

$$d(x_1, x_2) = \sqrt{(x_1 - x_2)^T (x_1 - x_2)} \tag{42}$$

La *Distancia de Mahalonobis* es más apropiada cuando las variables tienen correlaciones y varianzas diferentes pero también es más costosa computacionalmente.

3.4. Métodos de interpolación determinísticos

Los métodos de interpolación determinísticos generan los valores de estimación basándose en relaciones matemáticas pero no tienen en cuenta la incertidumbre de los datos.

3.4.1. Interpolador del vecino más próximo

Se trata de la solución más simple y consiste en asignar el valor del punto más cercano. Este método no se debe usar si los datos son ruidosos o si hay zonas del espacio a interpolar donde el número de puntos de control es reducido [64].

$$f(p) = y(min(d_i) \forall i = 1, ...N) \tag{43}$$

donde $d_i = d(p, x^i)$ que es la distancia Euclídea entre el punto a interpolar p y el i-ésimo punto de control.

Existe una variación de este mismo método denominado **k-vecinos más cercanos** (k-NN). Este método toma como valores en ubicaciones no maestreadas el promedio de las k ubicaciones cercanas. La fórmula que lo modela es:

$$\hat{Z}(s_0) = \sum_{i=1}^{k} \frac{Z(s_i)}{k} \tag{44}$$

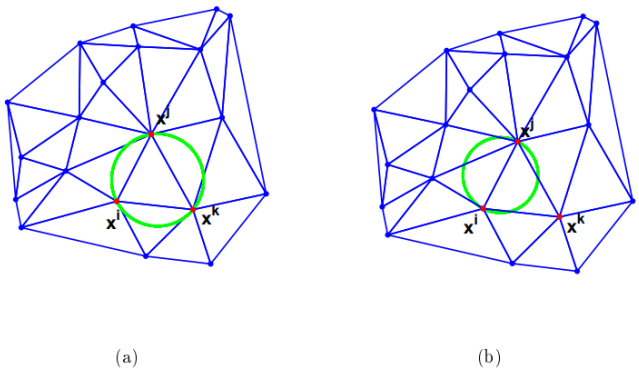
donde $\hat{Z}(S_0)$ es el valor predicho en s_0, $Z(s_i)$ será el valor observado correspondiente al vecino s_i, por último, k será el número de vecinos a considerar.

3.4.2. Triangulación lineal (TIN)

Se trata de un **interpolador local** y exacto, basado en la generación de una malla de triángulos irregulares de Delaunay cuyos vértices son los puntos de control. Siendo $X = x^n, n = 1, ..., N$ un conjunto de puntos de control, la triangulación T de Delaunay de X cumple las siguientes propiedades:

- Propiedad 1: tres puntos de X son vértices de la misma cara de un triángulo de Delaunay de X, si y solamente si, el círculo que pasa por ellos no contiene puntos de X en un interior. Véase Fig.35a

- Propiedad 2: dos puntos x^i, x^j pertenecientes a X definen un lado de un triángulo de Delaunay, si y solamente si, existe un círculo que contiene a x^i, x^j en su circunferencia y no contiene en su interior ningún punto de X. Véase Fig.35b



(a) (b)

Figura 35: Ilustración de las propiedades de la TIN (a) Primera propiedad (b) Segunda propiedad. Referencias:[64]

El valor a interpolar se determina a partir de un modelo de regresión lineal, teniendo en cuenta los tres vértices del triángulo $(x^1 = (x^1, t^1), x^2 = (s^2, t^2), x^3 = (s^3, t^3))$, éstos deben contener al punto $p = (s, t)$ a interpolar, y las correspondientes imágenes son $y^1 = f(x^1), y^2 = f(x^2), y^3 = f(x^3)$, el valor v a interpolar en p viene dado por:

$$f(p) = a + bs + ct \tag{45}$$

donde a, b y c son los coeficientes que definen la superficie plana de primer orden que se apoya en los vértices del triángulo de Delaunay. Estos coeficientes se obtienen del siguiente sistema de ecuaciones.

$$y^1 = a + bs^1 + ct^1 \tag{46}$$
$$y^2 = a + bs^2 + ct^2 \tag{47}$$
$$y^3 = a + bs^3 + ct^3 \tag{48}$$

Este tipo de método se adapta bien a los datos dispersos ya que no se depende de una malla regular. Por otro lado no genera valores fuera del rango original, a diferencia de otros métodos, de esta manera no producirá sobreajuste. Se trata de un método especialmente interesante para datos topográficos o geoespaciales.

3.4.3. Vecinos naturales

El método de los Vecinos naturales (*Natural Neighbors*), se trata de un método local, basado en un concepto de interpolación de datos dispersos definidos en función del diagrama de Voronoi, que además, se puede obtener a partir de los triángulos de Delaunay.

La idea del diagrama de Voronoi se basa principalmente en la proximidad. Suponiendo un conjunto finito de puntos en un plano $P = p_1, ..., p_n$, a cada p_j se le asocian los puntos en el plano que están más cerca o igual suya que cualquier otro punto p_i. De esta forma, todo punto en el plano quedará asociado con otro (p_i) formándose conjuntos que recubren este [65].

Además, también existirán puntos que disten lo mismo de los elementos de P y que formarán la frontera de cada región. Los conjuntos resultantes forman una teselación del plano, llamados estos *Diagrama de Voronoi* y, cada una de las regiones resultantes serán *Regiones de Voronoi* (ver Fig.36 [65].

La relación que se establece entre el diagrama de Voronoi y los triángulos de Delaunay se establece porque, el centro del círculo circunscrito por cada triángulo de Delaunay corresponderá con un vértice generador del diagrama de Voronoi como se muestra en la Fig.37, las perpendiculares a los lados del triángulo forman las aristas del diagrama de Voronoi [64].

Los pasos a seguir en este método será, primero, el cálculo de las coordenadas locales del punto de consulta (x_0) con respecto a sus vecinos naturales $N(x_0)$ [66]. A continuación, se realizará una combinación de valores de la siguiente manera:

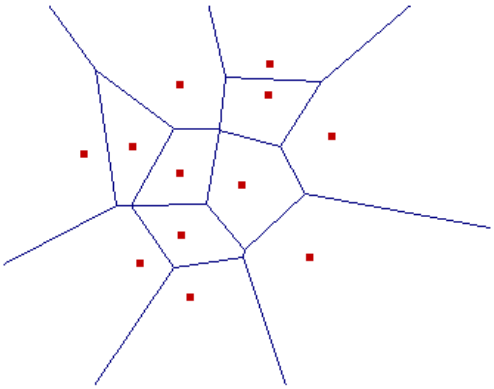


Figura 36: Representación gráfica del diagrama de Voronoi. Referencias: [65]

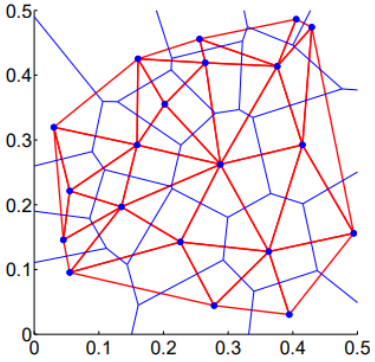


Figura 37: Relación entre la triangulación de Delaunay (rojo) y del correspondiente diagrama de Voronoi (azul). Referencias: [64]

$$f(x_0) = \sum_{i \in N(X_0)} \lambda_i z_i \tag{49}$$

donde λ_i serán las coordenadas de los vecinos naturales. Éstas se pueden determinar por las áreas o volúmenes de intersección entre los diagramas de Voronoi con y sin el punto de consulta de la siguiente manera [66]:

$$\lambda_i = \frac{A_{x_0}}{A_{Tx_0}} \tag{50}$$

donde A_{x_0} será el área del polígono de intersección con x_0 y, A_{Tx_0} el área total del polígono Voronoi de x_0 [66]. Esto garantiza lo siguiente:

$$\sum_i \lambda_i = 1; \lambda_i \geq 0 \tag{51}$$

Una de las principales ventajas de este tipo de interpolación es que, en este caso, no se tienen parámetros libres y, por tanto no será necesario a priori fijar el número de puntos de control a considerar en la interpolación, sino que éste será variable, dependiente de la configuración espacial de los puntos de control [64]

3.4.4. Ponderación inversa a la distancia (IDW)

Dentro de los interpoladores más usados para las interpolaciones espaciales están los métodos basados en las distancias inversas (IDW) junto con los que se verán a continuación *Splines* [11].

Los principios en los que se basa este método son [67]:

- **Vecindario de IDW:** se asume que la influencia de una variable disminuye co la distancia.
- **Control de la influencia mediante el parámetro de potencia** (u)**:** la interpolación dependerá de la distancia inversa elevada a una potencia matemática. De esta manera se puede ajustar la influencia de los puntos conocidos sobre los valores interpolados.
  - Valores altos dan más énfasis a puntos cercanos (menos suave).
  - Valores bajos permiten que los puntos más distantes tengan influencia y, de esta manera la superficie será más suave.
  - Valor recomendado generalmente será 2.

Este método hace uso de un algoritmo simple basado en las distancias [64]. De la siguiente manera:

$$f(p) = \frac{\sum_{i=1}^{n} \omega(d_i) y(x^i)}{\sum_{i=1}^{n} \omega(d_i)} \tag{52}$$

donde n será un parámetro que indica el número de puntos de control a tener en cuenta en la interpolación, d_i representa la distancia del i-ésimo punto de control más próximo a p. Por último, $\omega(d_i)$ será una función de ponderación que dependerá de la distancia entre el i-ésimo punto de control y p.

En cuanto a la función, ésta suele considerarse la siguiente:

$$\omega(d_i) = \frac{1}{(d_i)^u + \delta} \simeq (d_i)^{-u} \tag{53}$$

donde δ es una constante de regularización con el fin de evitar inestabilidades numéricas cuando el denominador está próximo a cero y, u será el denominado factor de ponderación, éste determina el modo en que la auto-correlación disminuye con el incremento de la distancia, habitualmente este es $u = 2$. Cuando menor sea u, más se asemejarán los pesos y más suave será la interpolación [64].

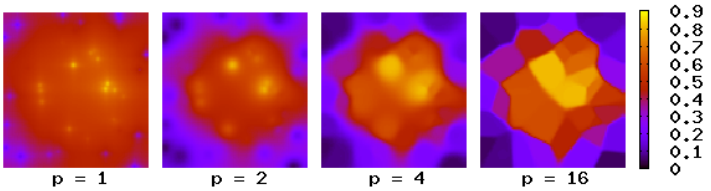


Figura 38: Variación de la interpolación con distintos valores de u. Referencias: [68]

La desventaja del uso de este método es a la hora de procesar grandes bases de datos, los tiempos serán grandes también en comparación con otros métodos [69]. Además, en este método será necesario la selección previa del parámetro n y, además, los resultados pueden estar influenciados por valores atípicos ya que no modela la variabilidad espacial.

3.4.5. Splines

La interpolación por Splines ajusta una función de forma fragmentaria haciendo uso de polinomios. Es decir, el rango a interpolar se divide en subintervalos no

solapados y, en cada uno de ellos se construye un polinomio generalmente diferente.

Se dividen en tres tipos diferentes:

- **Spline lineal:** hace uso de polinomios de grado 1 y, en general, su resultado no es suave ya que la primera derivada no será continua en los nodos.
- **Spline cuadrática:** hace uso de polinomios de grado 2 en cada uno de los intervalos, se impone ahora la continuidad de la función y su primera derivada.
- **Spline cúbico (el más común):** hace uso de polinomios de grado 3, garantiza continuidad de la función.

La fórmula en caso de los *Spline cúbicos* será la siguiente:

$$f(x) = \begin{cases} a_1x^3 + b_1x^2 + c_1x + d_1, & \text{si } x \in [x_1, x_2] \\ a_2x^3 + b_2x^2 + c_2x + d_2, & \text{si } x \in (x_2, x_3] \\ \vdots & \\ a_nx^3 + b_nx^2 + c_nx + d_n, & \text{si } x \in (x_n, x_{n+1}]. \end{cases} \tag{54}$$

Con una elección de coeficientes (a_i, b_i, c_i, d_i) para los polinomios, el resultado de la función pasará a través de los puntos de manera suave.

Para obtener los coeficientes el proceso será el siguiente:

- a. Cada polinomio deberá pasar a través de los puntos. Teniendo en cuenta que n será el número de puntos se deberán generar $2n$ ecuaciones.

$$S_i(x_i) = y_i \tag{55}$$

$$S_i(x_{i+1}) = y_{i+1} \tag{56}$$

- b. Condición de continuidad de la primera derivada: cada una de las primeras derivadas deberá coincidir en los puntos intermedios. Esto generará $n - 1$ ecuaciones:

$$S_i'(x_{i+1}) = S_{i+1}'(x_{i+1}) \tag{57}$$

- c. Condición de continuidad de la segunda derivada: deberá ocurrir lo mismo que en la primera, éstas, deberán coincidir en los puntos intermedios.

$$S_i''(x_{i+1}) = S_{i+1}''(x_{i+1}) \tag{58}$$

- d. Imponer las condiciones en los límites, en función de éstas hay diferentes categorías:

  - **Natural Spline:** la segunda derivada en los extremos es cero. Interpolación suave y natural.
  - **Not-a-Knot Spline:** la tercera derivada en el segundo y penúltimo punto sea igual. Transición más uniforme en la curvatura.
  - **Periodic Spline:** impone que las primeras y segundas derivadas sean iguales en los extremos. Útil en ecuaciones con comportamiento repetitivo.
  - **Quadratic Spline:** el primer y el último polinomio son de grado dos en lugar de tres. Se usa cuando la forma general de los datos permite un ajuste más simple.

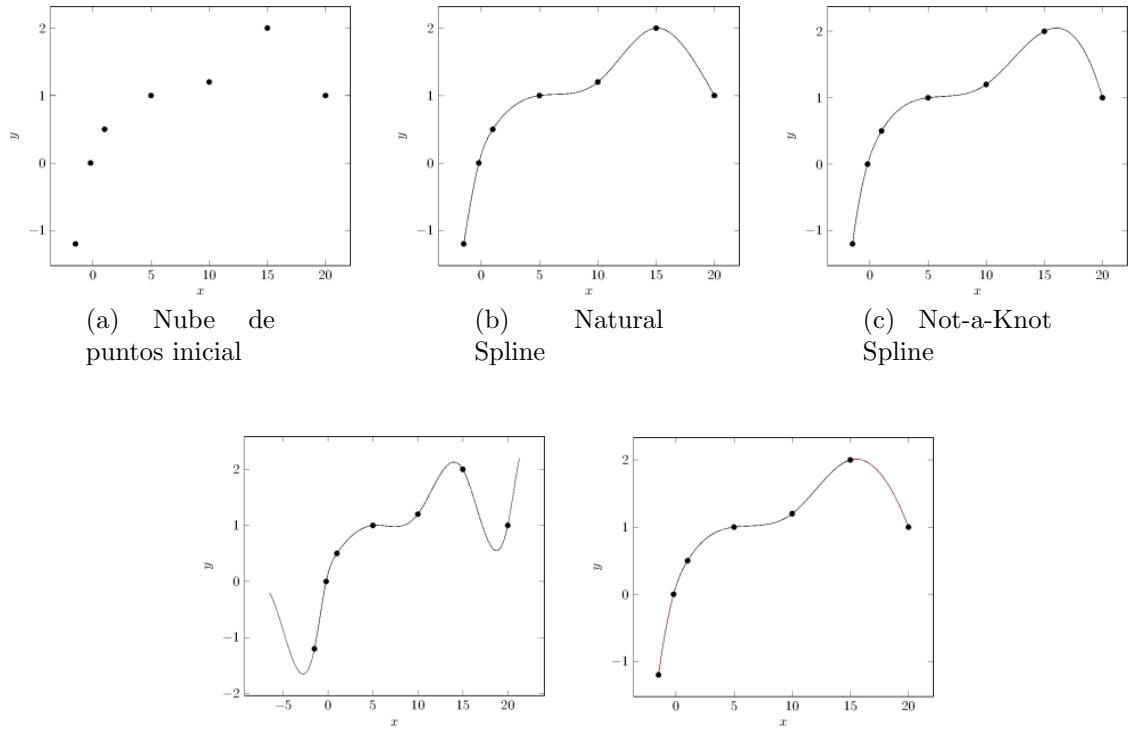
- e. Resolución del problema: el sistema resultante tendrá $4n$ ecuaciones con $4n$ incógnitas. Se puede resolver por el método de Gauss o mediante librerías en Python/R.

3.5. Modelos de interpolación geoestadísticos

Este tipo de métodos hacen uso de modelos estadísticos para optimizar las estimaciones considerando la relación espacial de los datos

3.5.1. Kriging ordinario (OK)

El Kriging Ordinario se trata de uno de los métodos más avanzados y utilizados en la geoestadísticos para la interpolación espacial de los datos. Es especialmente efectivo cuando la estructura de los datos es espacialmente compleja y no homogénea.



(a) Nube de puntos inicial

(b) Natural Spline

(c) Not-a-Knot Spline

(d) Periodic Spline

(e) Quadratic Spline

Figura 39: Modelos de condiciones límite. Referencias: [70]

A diferencia de métodos anteriormente nombrados como IDW o Splines, el Kriging no solo estima los valores en los puntos no muestreados, sino que también tiene en cuenta la dependencia espacial entre los puntos.

Hay numerosos tipos de Kriging, en este caso se valorará únicamente el Kriging Ordinario. En éste se creará una predicción para un punto del que no se tienen datos $\hat{Y}(x_{n+1})$ a partir de una combinación de los puntos de los que sí se tiene información x_i [71]:

$$\hat{Y}(x_{n+i}) = \sum_{i=1}^{n} \lambda_i \cdot Y(x_i) = \lambda' \cdot Y \tag{59}$$

donde $\sum_{i=1}^{n} \lambda_i = 1$, $\lambda = (\lambda_1, ..., \lambda_n)'$ y, $Y = (Y(x_1), ..., Y(x_n))'$. Este tipo de método asume una única salida para una combinación de entradas. En el caso de buscar numerosas salidas el predictor deberá ser computarizado para cada salida (Kriging multivariable [71]).

Para cuantificar los pesos λ hace uso del çriterio del sesgo", éste indica que, cuando predices el valor de un punto que ya has observado, el valor predicho debe ser exactamente igual al valor observado. Esto asegura que el Kriging es un interpolador exacto. Con el fin de saber la diferencia entre el dato real y el predicho se hace uso de diferentes tipos de métodos, éstas se denominan *Funciones de Correlación Espacial*.

Variograma y Semivariograma

El **variograma** se trata de una función que mide cómo varía una variable espacial en función de la distancia h entre puntos. Se denota como $2\gamma(h)$ y se basa en la suposición de que los incrementos de la variable tienen varianza infinita.

$$2\gamma(h) = E((Z(x+h) - Z(x)^2) \tag{60}$$

El **semivariograma** es únicamente la mitad del variograma $\gamma(h)$, y se usa para describir la dependencia espacial de los datos.

$$\bar{\gamma} = \frac{\sum(Z(x+h) - Z(x))^2}{2n} \tag{61}$$

donde $Z(x)$ será el valor de la variable en un sitio x y, $Z(x+h)$ en otro valor muestral separado una distancia h, n será el número de parejas que se encuentran separadas esa distancia.

Covariograma y Correlograma

La función de covarianza muestral se lleva a cabo haciendo uso de la covarianza muestral clásica:

$$C(h) = cov(Z(x+h), Z(x)) = \frac{\sum_{i=1}^{n}(Z(x+h) \cdot Z(x)}{n} - m^2 \tag{62}$$

donde m representa el valor promedio en todo punto de la región de estudio. Sin embargo, la única función que no requiere de hacer una estimación de parámetros es la *función de semi-varianza* o *semivariograma* por lo cual en la práctica se suele hacer uso de ella y no de las otras dos.

El modelo del semivariograma requiere de un ajuste de modelos que generalicen lo observado en el semivariograma a cualquier distancia. Existen diversos modelos teóricos de semi-varianza, éstos se pueden dividir en no acotados (lineal, logarítmico y potencial) y acotados (esférico, exponencial y gaussiano). Los del segundo grupo garantizan que la covarianza de los incrementos es infinita. Todos los modelos tienen tres parámetros en común:

- **Efecto pepita:** se denota por C_o y representa una discontinuidad puntual.
- **Meseta:** cota superior del semivariograma.Se denota como C_1. Está referida a la parte curva donde la semi-varianza se estabiliza. Indica que, a partir de cierta distancia, los puntos ya no están correlacionados espacialmente, lo que significa que no hay más dependencia espacial entre ellos.

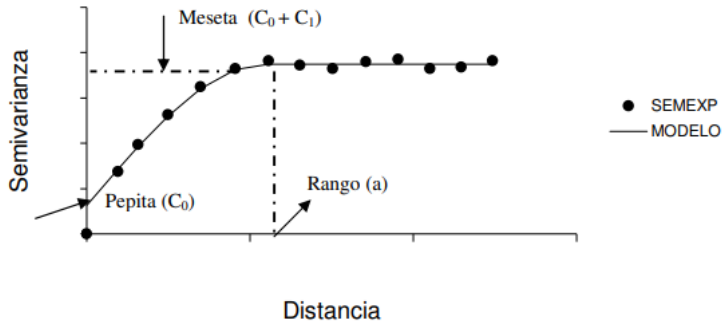


Figura 40: Comportamiento típico de un semivariograma acotado. Referencias: [72]

- **Rango:** distancia a partir de la cual dos observaciones son independientes.Denotado como a.

A continuación se expondrán brevemente los modelos teóricos de semi-varianza [72]:

- **Modelo esférico:** crecimiento rápido cerca del origen.

$$y(x) = \begin{cases} C_0 + C_1(\frac{3}{2}(\frac{h}{a})\frac{1}{2}(\frac{h}{a})^3, & \text{si } h \leq a \\ C_0 + C_1, & \text{si } h > a \end{cases} \tag{63}$$

- **Modelo exponencial:** cuando la dependencia espacial tiene un crecimiento exponencial respecto a la distancia.

$$y(h) = C_0 + C_1(1 - exp(\frac{-3h}{a})) \tag{64}$$

- **Modelo Gaussiano:** la dependencia espacial se desvanece solo en una distancia que tiende a infinito.

$$y(h) = C_0 + C_1(1 - exp(\frac{-h^2}{a^2})) \tag{65}$$

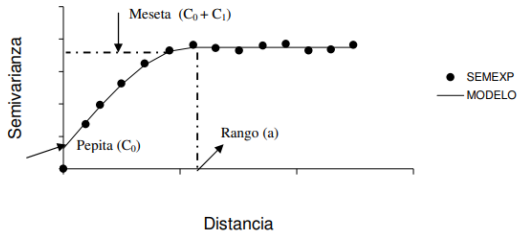


Figura 41: Comparación de los modelos exponencial, esférico y Gaussiano. Referencias: [72]

En la Fig.42 se muestra el método de Kriging en relación con la regresión polinómica de segundo orden.

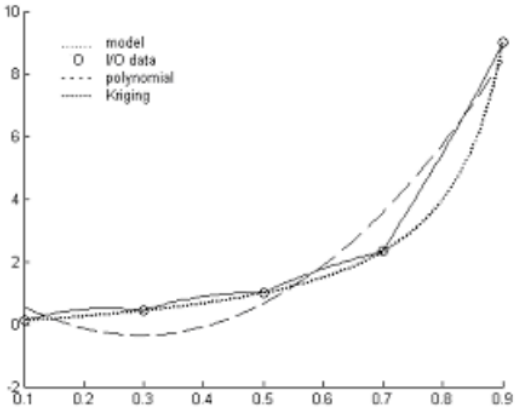


Figura 42: Método de Kriging en relación con la regresión polinómica de segundo orden. Referencias: [71]

En este método se hace uso de técnicas con el fin de obtener los puntos de entrada más relevantes de un modelo de simulación. El principal será **Método LHS**, éste distribuye los puntos de manera más equitativa en todo el espacio. Es decir, los

puntos de entrada se eligen de forma que cubra mejor todo el rango posible de valores de entrada [71].

En la Fig.43 se muestra un ejemplo con k = 2. El proceso seguido para construir este diseño será [71]:

- LHS dividirá cada rango de entrada en n intervalos de igual longitud, nombrados del 1 al n (por lo que el número de valores por entrada puede ser mucho mayor que en los diseños para polinomios de bajo orden).
- LHS colocará estos números de manera que cada uno de ellos aparezca exactamente una vez en cada fila y cada columna de la matriz de diseño.
- Dentro de cada celda de la matriz, el valor exacto de entrada puede ser muestreado uniformemente. (Alternativamente, estos valores pueden colocarse en el centro de cada celda).

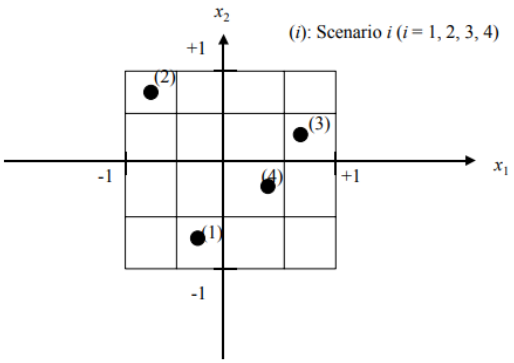


Figura 43: Diseño LHS para dos factores y cuatro escenarios. Referencias: [71]

En resumen el Kriging Ordinario se trata de un potente método para la estimación de variables espaciales pero supone una gran complejidad computacional y además, depende en gran medida de la calidad y precisión del modelo de variograma (covarianza).

3.6. Modelos de interpolación avanzados (redes neuronales)

La interpolación de datos espaciales tradicionalmente ha hecho uso de herramientas como las comentadas anteriormente para estimar valores en ubicaciones no maestreadas. Sin embargo, estos métodos pueden presentar limitaciones cuando las

relaciones entre variables son altamente no lineales o cuando los datos presentan una gran variabilidad espacial. Las redes neuronales tienen la capacidad de comprender patrones complejos, aprender de los datos y generalizar mejor en presencia de factores independientes. Un mapa lineal presenta una relación entre la entrada de los datos y la salida, mantiene las proporciones de los datos, son fáciles de calcular y entender y no pueden capturar relaciones complejas entre los datos. En el caso de mapas no lineales la relación entre la entrada y la salida no sigue una relación fija.

Este método se trata del más simple que, en el caso de en términos de redes neuronales tradicionales se trata de un método con una sola capa oculta y una capa de salida lineal o *perceptrón multicapa*. Por tanto, estarán modeladas por las ecuaciones 15 y 20. En el cálculo de interpolación de mapas puede ser interesante estudiar la complejidad (número de parámetros) determinada por el número de neuronas ocultas (k), entradas (n) y salidas (m), y está representada por la fórmula [73]:

$$N_c = k(n + m + 1) + m \tag{66}$$

La complejidad se basa en el número de parámetros que necesitan ser ajustados para aproximar un mapeo específico.

Por otro lado, también aparecieron métodos con funciones base radiales (RBF) con el fin de dar solución a problemas de interpolación en varias variables. Las redes neuronales de base radial se basan en la ecuación:

$$z = \sum_{i=1}^{k} \omega_i \dot{\phi}(||x - c_i||) + b \tag{67}$$

donde ϕ es la función de base racial, c_i los centros de las funciones radiales y $||x - c_i||$ será la distancia euclidiana entra la entrada x y el centro c_i.

Un estudio ha comparado métodos geoestadísticos como el kriging con métodos basados en redes neuronales (perceptrón multicapa y redes neuronales de función de base radial). Los resultados se muestran en las Fig.45 y 46.

En las figuras anteriores se puede observar que el Kriging es ideal cuando se tiene un conocimiento perfecto de la variabilidad espacial, pero las redes neuronales, pue-

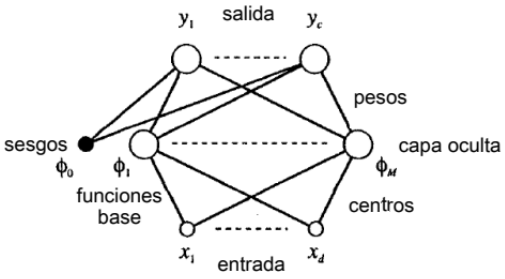


Figura 44: Arquitectura de una red neuronal de funciones base radiales. Referencias: [74]

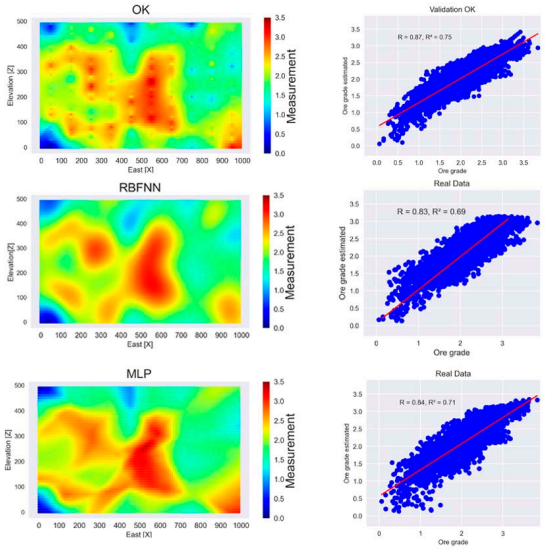


Figura 45: Comparación entre el uso del método Kriging Ordinario, perceptrón multicapa y redes neuronales de función de base radial. Referencias: [75]

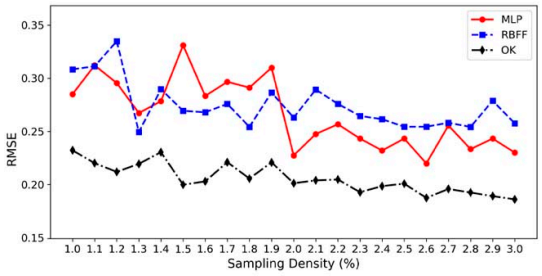


Figura 46: Comparación de variación de la densidad de muestreo y los distintos métodos. Referencias: [75]

den acercarse correctamente a estos resultados. El estudio [75] también indica que el perceptrón multicapa alcanzó buenas métricas después de ajustar los hiperparámetros, la red de base radial fue más eficiente en el tiempo. Por otro lado, la densidad de muestreo se puede observar que juega un papel importante. Al variar el tamaño

de la muestra entre el 1 % y el 3 % los resultados continuaron siendo consistentes. Esto sugiere que el método puede ser robusto a tamaños de muestra y, también indica aun así la importancia de la disposición espacial ya que puede influir de algún modo en los resultados.

4. Análisis e ingesta de la base de datos

En el siguiente apartado se realizará una descripción del proceso de recopilación, preparación e incorporación de los datos. Se detallarán las características de estos así como las técnicas empleadas para la limpieza es éstos. Se evaluará la calidad y los posibles datos faltantes describiendo así el preprocesamiento aplicado.

Este paso resultará fundamental para una correcta interpretación de los datos a la hora de analizarlos.

4.1. Datos de contaminantes

La base de datos de contaminantes utilizada en el presente proyecto cuenta con datos desde 2008 hasta 2019, además, cada uno de estos años se ha recopilado información sistemática relacionada con los principales contaminantes que afectan la calidad del aire, ya mencionados en apartados anteriores. Estos datos son: monóxido de carbono (CO), el dióxido de nitrógeno (NO_2), partículas sólidas en suspensión con un diámetro aerodinámico inferior a 10 [μm] y 2,5 [μm] (PM_{10}, $PM_{2.5}$), dióxido de azufre (SO_2) y por último, ozono troposférico (O_3). Cada uno de estos contaminantes se encontrarán representados mediante diferentes variables que reflejan su concentración medida en diferentes estaciones, ubicaciones o condiciones, de esta manera se podrá realizar un análisis detallado tanto a nivel espacial como temporal.

Estos datos se obtienen en formato .dat por lo que el primer paso será interpretar y estructurar los datos en la programación RStudio. Con este fin se han desarrollado el código 1.

```
carpeta <- "Directorio donde se encuentran los archivos"

# Obtener todos los archivos .DAT en la carpeta
archivos_dat <- list.files(carpeta, pattern = "\\.dat$", full.names =
    TRUE)

# Ver los primeros archivos listados
head(archivos_dat)

# Leer todos los archivos .dat y almacenarlos en variables con sus
    nombres
for (archivo in archivos_dat) {
```

```
  nombre_base <- tools::file_path_sans_ext(basename(archivo))

  # Leer el archivo
  df <- read.table(archivo, header = FALSE, sep = "", stringsAsFactors
      = FALSE)

  # Renombrar las columnas
  colnames(df) <- c("Longitud", "Latitud", "CONCENTRACIN")

  # Asignar la tabla a una variable con el nombre del archivo
  assign(nombre_base, df, envir = .GlobalEnv)
}

rm(archivo,
   archivos_dat,
   carpeta,
   nombre_base)


# Renombrar variables
NO2 <- Integrado_por_poblacion_mapa_final_NO2_bias_anual_blank_new
PM10 <- Integrado_por_poblacion_mapa_final_PM10_bias_anual_blank_new
PM25 <- Integrado_por_poblacion_mapa_final_PM25_bias_anual_blank_new
SO2 <- Integrado_por_poblacion_mapa_final_SO2_bias_anual_blank_new
CO <- `conc-anual-CO_mod_maxoctoh_2019_blank`
O3_1 <- mapa_diario_O3_26th_2019_nueva_metodologia_blank
O3_26 <- mapa_O3_1th_2019_nueva_metodologia_blank

# Eliminar variables originales completamente
rm(Integrado_por_poblacion_mapa_final_NO2_bias_anual_blank_new,
   Integrado_por_poblacion_mapa_final_PM10_bias_anual_blank_new,
   Integrado_por_poblacion_mapa_final_PM25_bias_anual_blank_new,
   Integrado_por_poblacion_mapa_final_SO2_bias_anual_blank_new,
   `conc-anual-CO_mod_maxoctoh_2019_blank`,
   mapa_diario_O3_26th_2019_nueva_metodologia_blank,
   mapa_O3_1th_2019_nueva_metodologia_blank,
   df)
```

Listing 1: Código para importar los datos de los ficheros .dat

El siguiente proceso se ha llevado con únicamente los datos del año 2019. El objetivo de esta decisión es realizar pruebas preliminares de interpolación espacial

con una muestra limitada, lo que permite evaluar la eficacia y adecuación de los diferentes métodos disponibles. Una vez se haya escogido el método más adecuado se llevará a cabo este proceso con el resto de años comprendidos en la base de datos.

Además, para poder estudiar la disposición de los datos resulta de interés realizar una visualización de estos en el mapa, para ello y, debido a que todos los contaminantes cuentan con los mismos puntos de evaluación se ha tenido en cuenta únicamente el contaminante *CO*. Con este fin, se ha hecho uso de las librerías *ggplot2* y *maps*. El código mostrado en 2 hace referencia a este proceso. Los resultados de aplicación de este código se muestran en la Fig.47.

```
# Cargar librer as
library(ggplot2)
library(maps)

# Crear un mapa base y aniadir los puntos de una tabla
ggplot() +
  borders("world", colour = "gray70", fill = "gray90") +
  geom_point(data = CO, aes(x = Longitud, y = Latitud), color = "red",
      alpha = 0.6) +
  # Ajustar para enfocarse en Espania
  coord_cartesian(xlim = c(-10,5),
                  ylim = c(30, 50)) +
  theme_minimal() +
  labs(title = "Ubicaciones de Puntos de C O ", x = "Longitud", y = "
      Latitud")
```

Listing 2: Código para crear un mapa con puntos de CO

Con la representación mostrada en la Fig.47 se puede analizar que los datos se encuentran repartidos entre las latitudes 35.55 y 44.70. También en las longitudes -10.45 y 5.50.

Posterior a esto se estudiaron los datos y se reparó en que muchos de los datos tenían valores de 1.70141×10^{38} lo cual resultan incoherentes con las concentraciones esperadas de cualquiera de estos contaminantes. Esto llevó a la conclusión de que, en el proceso de almacenamiento, dichos valores probablemente correspondían a datos nulos que, por algún error o sistema propio de numeración, fueron representados de esta manera. Con el fin de que estos números no dieran problemas en el futuro se procedió a catalogarlos como *NaN* tal y como se muestra en el código 3. Además, como se puede observar también se escogieron los datos válidos y se incluyeron en

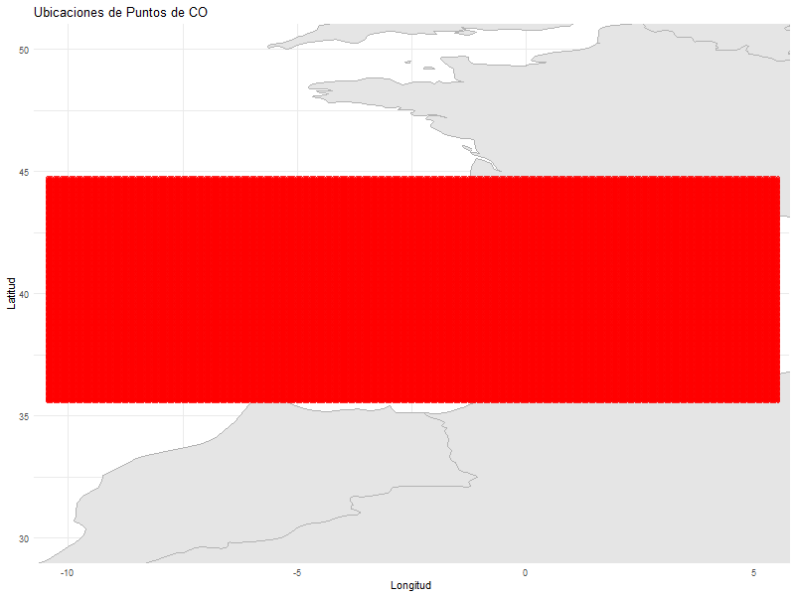


Figura 47: Mapa de los datos de monóxido de carbono sin filtrar. Referencias: *elaboración propia.*

variables independientes indicadas con el prefijo VV_* referidos a *Valores Válidos*.

```
# Filtrar valores 1.70141E+038 -> NaN
CO$CONCENTRACIN[CO$CONCENTRACIN == 1.70141E+038] <- NaN
NO2$CONCENTRACIN[NO2$CONCENTRACIN == 1.70141E+038] <- NaN
O3_1$CONCENTRACIN[O3_1$CONCENTRACIN == 1.70141E+038] <- NaN
O3_26$CONCENTRACIN[O3_26$CONCENTRACIN == 1.70141E+038] <- NaN
PM10$CONCENTRACIN[PM10$CONCENTRACIN == 1.70141E+038] <- NaN
PM25$CONCENTRACIN[PM25$CONCENTRACIN == 1.70141E+038] <- NaN
SO2$CONCENTRACIN[SO2$CONCENTRACIN == 1.70141E+038] <- NaN

# Filtrar las filas donde CONCENTRACIN es mayor o igual a 0 y no es
    NA -> valores v lidos de CO
VV_CO <- CO[CO$CONCENTRACIN >= 0 & !is.na(CO$CONCENTRACIN), ]
VV_NO2 <- NO2[NO2$CONCENTRACIN >= 0 & !is.na(NO2$CONCENTRACIN), ]
VV_O3_1 <- O3_1[O3_1$CONCENTRACIN >= 0 & !is.na(O3_1$CONCENTRACIN),
    ]
VV_O3_26 <- O3_26[O3_26$CONCENTRACIN >= 0 & !is.na(O3_26$
    CONCENTRACIN), ]
VV_PM10 <- PM10[PM10$CONCENTRACIN >= 0 & !is.na(PM10$CONCENTRACIN),
    ]
VV_PM25 <- PM25[PM25$CONCENTRACIN >= 0 & !is.na(PM25$CONCENTRACIN),
    ]
VV_SO2 <- SO2[SO2$CONCENTRACIN >= 0 & !is.na(SO2$CONCENTRACIN), ]
```

```
# Resumen de los datos
summary(VV_CO)
summary(VV_NO2)
summary(VV_O3_1)
summary(VV_O3_26)
summary(VV_PM10)
summary(VV_PM25)
summary(VV_SO2)
```

Listing 3: Código para el filtrado de los datos nulos en los contaminantes.

A continuación se presentan los diferentes resúmenes estadísticos de las contaminantes filtrados. Para cada contaminante se incluyen los valores mínimos, máximos, primer cuartil (Q1), mediana, tercer cuartil (Q3) y la media. Se muestran estos resultados en las Tablas 3 y 4.

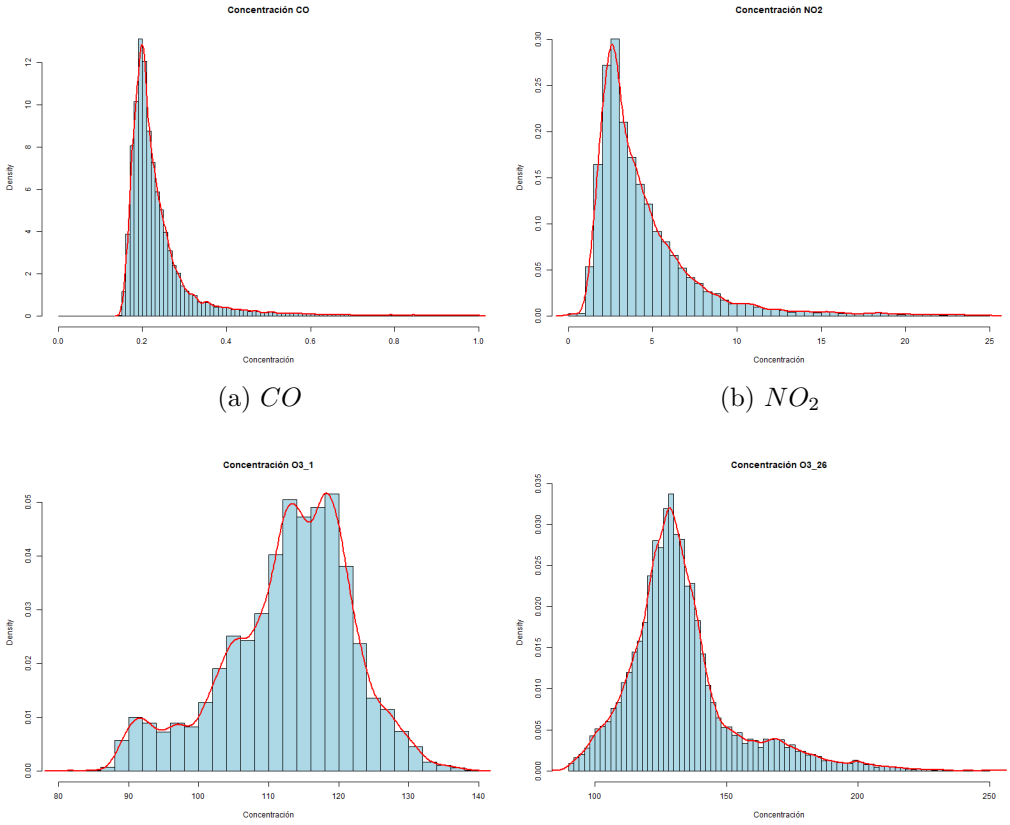
Contaminante	**Mínimo**	**1er Cuartil** (Q1)	**Mediana**	**Media**
CO	0.1500	0.1920	0.2120	0.2448
NO	0.012	2.514	3.507	4.682
O (1h)	79.22	107.60	114.00	112.86
O (26h)	83.53	120.90	129.50	132.98
PM	8.125	11.640	12.740	13.288
PM	4.506	6.496	7.095	7.391
SO	0.299	2.010	2.123	2.352

Tabla 3: Resumen estadístico de concentraciones de contaminantes (parte 1)

Contaminante	**3er Cuartil** (Q3)	**Máximo**
CO	0.2510	5.2140
NO	5.368	92.460
O (1h)	119.10	138.50
O (26h)	139.40	257.80
PM	14.450	34.420
PM	7.991	27.220
SO	2.362	41.510

Tabla 4: Resumen estadístico de concentraciones de contaminantes (parte 2)

Además, con el fin de poder estudiar más en profundidad el conjunto de los datos se optó por desarrollar histogramas (Fig.48 y 49) de cada uno de ellos y así poder analizar su forma.



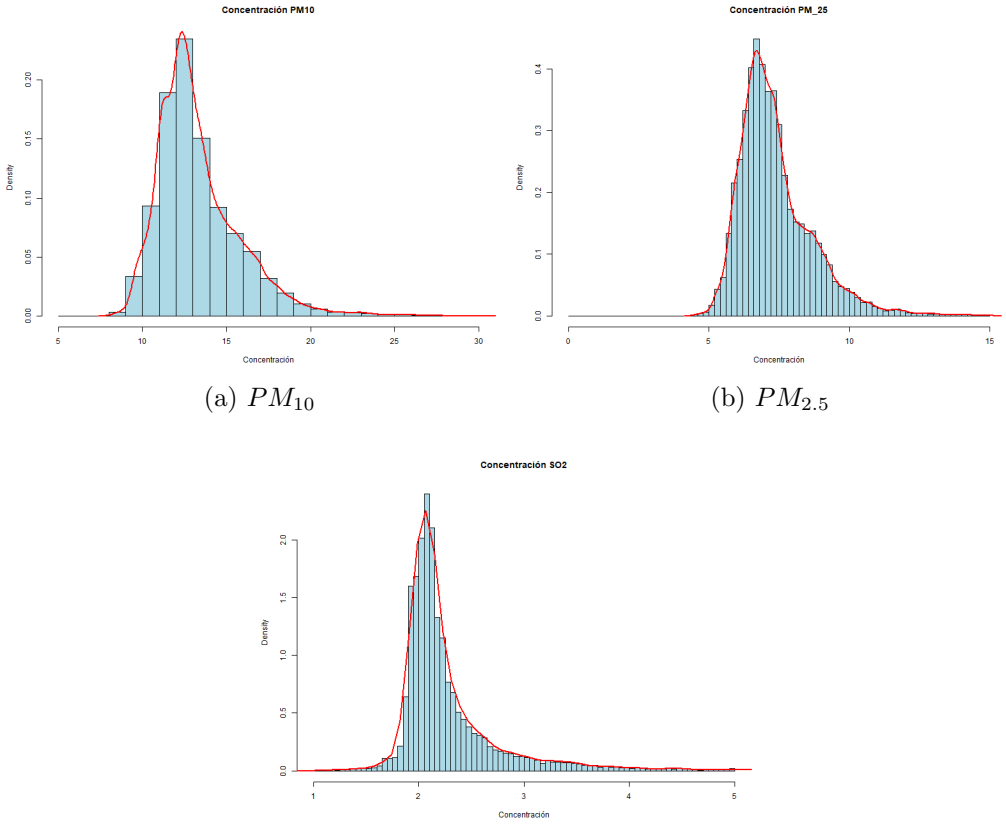
(a) CO (b) NO_2

(c) $O_3(1h)$ (d) $O_3(26h)$

Figura 48: Histogramas de las concentraciones de los diferentes contaminantes (Parte 1). Referencias: elaboración propia

Teniendo en cuenta los diferentes tipos de distribuciones mostradas en la Fig.50 se han clasificado las distribuciones de los tipos de contaminantes en el año 2019 de la siguiente manera:

- **Distribución logarítmica normal:** Las distribuciones que parecen mostrar un comportamiento similar al de una distribución logarítmica normal el monóxido de carbono (Fig.48a), el dióxido de nitrógeno (Fig.48b) y las partículas sólidas en suspensión de menos de 10 $[\mu m]$ y 2.5 $[\mu m]$ (Fig.49a y Fig.49b) y el dióxido de azufre (49c).
- **Distribución normal:** La concentración de ozono medida en intervalos de 1 hora (Fig.48c)
- **Distribución *t de Student*:** El caso del ozono medido a lo largo de 26 horas (Fig.48d).



(a) PM_{10} (b) $PM_{2.5}$

(c) SO_2

Figura 49: Histogramas de las concentraciones de los diferentes contaminantes (Parte 2). Referencias: elaboración propia

La obtención de los histogramas se ha llevado a cabo siguiendo el código 4. Este código es únicamente para el histograma de CO. Se ha aplicado este mismo para el resto de contaminantes.

```
14      png("Directorio de almacenamiento", width = 800, height = 600)
# Filtrar valores
16 datos_filtrados <- VV_CO$CONCENTRACIN[VV_CO$CONCENTRACIN >= 0 & VV_
       CO$CONCENTRACIN <= 1]
# Crear histograma con esos valores
18 hist(datos_filtrados,
          main = "Concentraci n CO",
20        xlab = "Concentraci n",
          col = "lightblue",
22        border = "black",
          probability = TRUE,
24        xlim = c(0, 1),
```

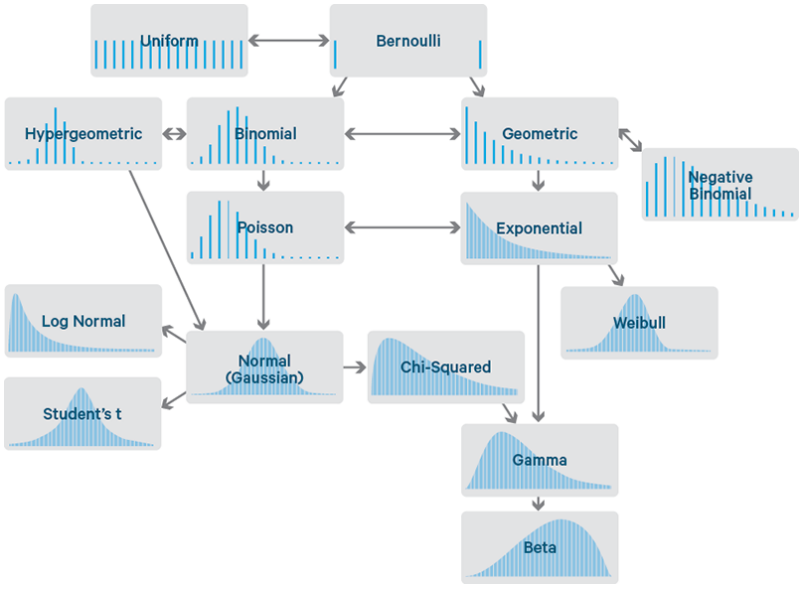


Figura 50: Tipos de distribuciones de probabilidad. Referencias: [76].

```
    breaks = seq(0, 1, by = 0.01))
# A adir la curva de densidad
lines(density(datos_filtrados), col = "red", lwd = 2)
dev.off()
```

Listing 4: Código para la obtención de histogramas de cada uno de los contaminantes.

Una vez haber finalizado el proceso de limpieza de los datos, se ha vuelto a hacer una representación de éstos sobre el mapa para poder estudiar de forma visual si se han limpiado de forma efectiva. Esta nueva visualización permite identificar posibles valores atípicos residuales, errores de localización o inconsistencias que hayan podido persistir tras la limpieza inicial. Esta representación se muestra en la Fig.51.

Como se puede observar en la Fig.51 y, comparándola con la referencia inicial (Fig.47) se puede llegar a la conclusión que los datos eliminados son aquellos que correspondían a puntos sobre el mar, Portugal, etc. Es decir, los datos que con mayor probabilidad, son nulos.

El código para dibujar esta gráfica se muestra en List.5.

```
   png("Directorio de descarga", width = 800, height = 600)
ggplot() +
  borders("world", colour = "gray70", fill = "gray90") +  # Mapa
      mundial
  geom_point(data = VV_CO, aes(x = Longitud, y = Latitud), color = "red
```

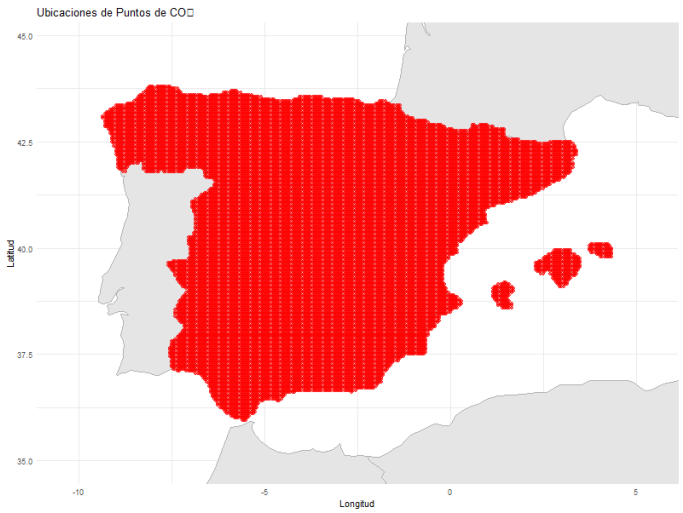


Figura 51: Resultado gráfico de la limpieza de los datos. Referencias: elaboración propia

```
    ", alpha = 0.6) +
  coord_cartesian(xlim = c(-10.35,5.35),
34                  ylim = c(34.95, 44.8)) +
  theme_minimal() +
36  labs(title = "Ubicaciones de Puntos de CO", x = "Longitud", y = "
      Latitud")
dev.off()
```

Listing 5: Código para dibujar los datos válidos sobre el mapa.

4.2. Datos de salud

Para este apartado se parte de una base de datos principal obtenida mediante diversos estudios clínicos a lo largo de los años de la población andaluza. La base de datos cuenta con 49 variables obtenidas de 823.276 personas repartidas por toda la provincia. Las variables con las que se cuenta son:

- Código postal.
- Sexo.
- Edad.
- Código y nombre del centro de salud.

- Código y nombre del distrito sanitario.
- Provincia.
- Código y nombre de financiación.
- Código y nombre del tramo farmacéutico.
- Porcentaje de pago de medicamentos.
- Presenta **diabetes** y fecha de inicio.
- Presenta **dislipemia** y fecha de inicio.
- Presenta **hipotiroidismo** y fecha de inicio.
- Presenta **obesidad** y fecha de inicio.
- Presenta **dependencia del tabaco** y fecha de inicio.
- Presenta **trastorno del estado de ánimo** y fecha de inicio.
- Presenta **trastorno de ansiedad** y fecha de inicio.
- Presenta **trastorno de la conducta alimentaria** y fecha de inicio.
- Presenta **enfermedad cardiovascular aguda** y fecha de inicio.
- Presenta **enfermedad cardiovascular mal definida** y fecha de inicio.
- Presenta **isquemia cerebral transitoria** y fecha de inicio.
- Presenta **secuela de enfermedad cerebro-vascular** y fecha de inicio.
- Presenta **arteriopatía de extremidades** y fecha de inicio.
- Presenta **arteriopatía intraabdominal** y fecha de inicio.
- Presenta **cardiopatías isquémicas** y fecha de inicio.
- Presenta **hipertensión** y fecha de inicio.
- Presenta **insuficiencia cardíaca** y fecha de inicio.
- Presenta **esteatosis hepática** y fecha de inicio.

Entre todas estas variables se procederá a eliminar algunas no relevantes para el estudio como lo son: código y nombre del centro de salud, código y nombre del distrito sanitario, código y nombre de financiación, nombre del tramo farmacéutico, porcentaje de pago de medicamentos (Lst. 8).

Hay que tener en cuenta también el significado de las variables dentro del tramo farmacéutico donde:

- **TSI 001**: Pensionistas con renta inferior a 18.000€/año.
- **TSI 002**: Pensionistas con renta entre 18.000 y 100.000€/año.
- **TSI 003**: Pensionistas con renta superior a 100.000€/año.
- **TSI 004**: Activos con renta inferior a 18.000€/año.
- **TSI 005**: Activos con renta entre 18.000 y 100.000€/año.
- **TSI 006**: Activos con renta superior a 100.000€/año.

Por otro lado, aunque algunas patologías podrían no presentar, a priori, una relación directa con los niveles de contaminación ambiental, se ha optado por no excluir ninguna categoría diagnóstica en las fases iniciales del análisis. Esta decisión responde al objetivo de evitar la introducción de sesgos derivados de juicios a priori o percepciones humanas no fundamentadas en evidencia empírica. Al mantener una aproximación exploratoria amplia, se favorece la detección de posibles correlaciones no anticipadas y se preserva la objetividad del estudio en las etapas de modelado y análisis estadístico. La depuración de variables no significativas se realizará posteriormente, con base en criterios estadísticos y metodológicos definidos.

También se cuenta con otra base de datos, se trata de una base de datos que cuenta con los centroides en latitud y longitud de los diferentes códigos postales. Esta base de datos cuenta con 1554 códigos postales diferentes y ocho variables: identificador, identificador del centro de salud, código postal, fecha de alta en la base de datos, provincia, capa, coordenada en x y coordenada en y. Además, en la misma base de datos se especifica el sistema de referencia espacial usando: EPSG:4258 / ETRS89.

De esta última base de datos solo se podrá especial interés en el código postal, longitud (xcoord) y latitud (ycoord).

El código para la importación de esta base de datos se encuentra en Lst. 6.

```r
library(dplyr)
library(readxl)

dat.salud <- read.csv("Directorio del archivo", sep = ";", header =
    TRUE)
Centroides.CP <- read_excel("Directorio del archivo")

CP <- Centroides.CP %>%
  select(COD_POSTAL, xcoord, ycoord)%>%
  rename(
    Longitud = xcoord,  # Nuevo nombre para xcoord
    Latitud = ycoord
)
```

Listing 6: Código para la importación de las bases de datos correspondientes.

El siguiente paso sería la localización de los códigos postales en la base de datos de salud con los códigos postales proveniente de la base de datos de centroides. El fin es poder incluir en la base de datos de parámetros de salud los datos correspondientes a latitud y longitud de cada uno de los puntos de muestra. Para ello se hizo uso de la función *merge*, esta función une dos data-frames basándose en una o más columnas comunes. Para ello, prestando atención a los tipos de datos que se poseían se pudo comprobar que ambos se encontraban en formatos diferentes. Para ello se forzó el formato tal y como se muestra en Lst. 7.

```r
dat.salud$COD_POSTAL <- sprintf("%05d", as.integer(dat.salud$COD_POSTAL
    ))
CP$COD_POSTAL <- sprintf("%05d", as.integer(CP$COD_POSTAL))
```

Listing 7: Cambio de formato en cada base de datos.

Además, observando el comportamiento de la función *merge* se pudo observar que, debido a que en la base de datos de centroides algunos códigos postales se repetían, la base de datos de parámetros de salud cambiaba. Por tanto, fue necesario eliminar aquellos códigos postales duplicados para finalmente poder aplicar la función. El código en que se aplica este proceso se muestra en Lst. 8.

```r
# Eliminar duplicados
CP <- CP[!duplicated(CP$COD_POSTAL), ]

dat.salud <- merge(dat.salud, CP[, c("COD_POSTAL", "Longitud", "Latitud
    ")], by = "COD_POSTAL", all.x = TRUE)
```

```
codigos_no_encontrados <- dat.salud$COD_POSTAL[!dat.salud$COD_POSTAL %
    in% CP$COD_POSTAL]

dat.salud$COD_CAP <- NULL
dat.salud$DESC_CAP <- NULL
dat.salud$COD_DISTRITO <- NULL
dat.salud$DESC_DISTRITO <- NULL
dat.salud$COD_FINANCIACION <- NULL
dat.salud$DESC_FINANCIACION <- NULL
dat.salud$COD_TRAMO_FARMACIA <- NULL
dat.salud$DESC_TRAMO_FARMACIA <- NULL
dat.salud$NUM_PORC_TRAMO_FARMACIA <- NULL
```

Listing 8: Aplicación de la función *merge*

Por último, como se puede observar, se ejecutó un código que permitiera saber que CP no habían sido encontrados. Estos datos se han aportado a mano a la base de datos haciendo uso de la herramienta *Geonames* para obtener la latitud y longitud correspondientes. De esta manera, ya se podrá trabajar con la base de datos de *dat.salud*, ya que esta ahora cuenta con el mismo formato de localización espacial con el que se contaba en el apartado 4.1.

Con la intención de poder analizar visualmente los datos de los que se partía, se realizó una gráfica en la que se muestran los diferentes puntos de toma de datos distribuidos en el mapa. La figura a la que se hace referencia es Fig.52. Como se puede observar en ella, los datos se encuentran bien distribuidos a lo largo del territorio andaluz.

Por otro lado, también se ha considerado de interés crear gráficas que permitan conocer que distribución estadística de datos tenemos. Para ello se ha realizado un diagrama circular con el porcentaje de enfermedades presentes en la base de datos (Fig.53a). También se ha generado una gráfica otros datos que podrían ser de interés como la edad (Fig.53b) y el sexo (Fig.53c).

Analizando estas tres gráficas se puede obtener como conclusiones:

- La mayoría de personas estudiadas no cuenta con ninguna enfermedad.
- Las enfermedades predominantes en la base de datos son hipotiroidismo e hipertensión.

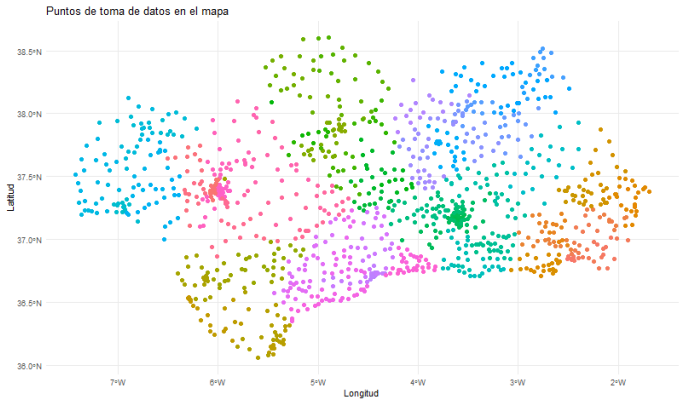
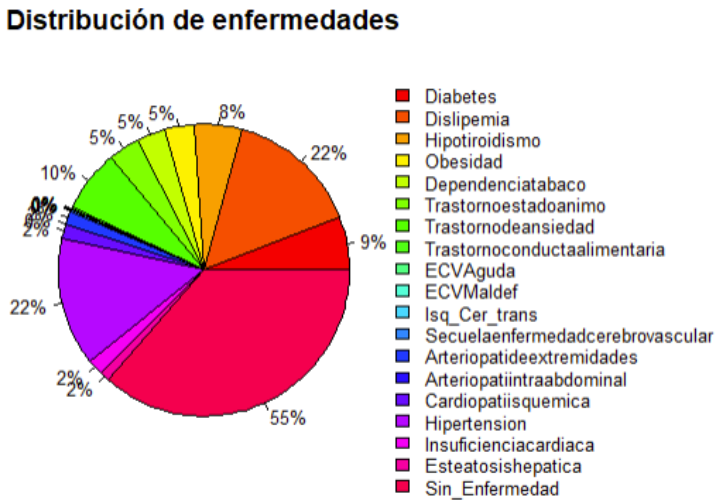
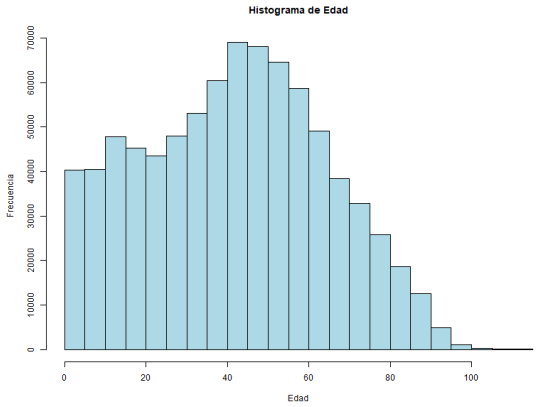


Figura 52: Distribución de la base de datos en el mapa. Referencias: elaboración propia

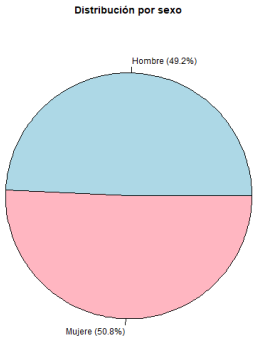
- El rango de edad del estudio es amplio teniendo un pico máximo en el rango entre 40 y 50 años.
- El sexo está equitativamente distribuido.



(a) Enfermedades



(b) Edades



(c) Sexo

Figura 53: Gráficas para la interpretación de la base de datos. Referencias: elaboración propia

5. Comparación y aplicación de los métodos de interpolación

Una vez preparados los datos, para el estudio el primer paso es el análisis de los diferentes tipos de interpolación estudiados. Para ello, se han aplicado estos métodos a una misma base de datos, con el fin de poder estudiar la viabilidad y la confiabilidad de éstos para su uso en el proyecto. A continuación se narran los diferentes tipos de interpolación aplicados y los resultados obtenidos de cada uno de ellos.

Los datos de contaminantes están repartidos en una cuadrícula a lo largo del territorio español, esto quiere decir que los puntos se encuentran repartidos de manera regular. Esto, en principio, beneficiará aquellos métodos que dependan muy directamente de proximidad entre los puntos.

Los datos no necesariamente serán lineales. Los focos de contaminación a lo largo del territorio pueden generar patrones no lineales ya que la concentración no dependerá directamente ni de la latitud ni de la longitud sino más bien de la distancia a los diferentes focos. Teniendo en cuenta lo estudiado en los apartados 3.4, 3.5 y 3.6 ciertos métodos pueden verse beneficiados y otros perjudicados por el tipo de datos con los que se cuentan.

- El método de **Vecinos cercanos o naturales** depende fuertemente de la densidad y distribución de los datos. Teniendo en cuenta la regularidad de estos, las decisiones pueden ser representativas. El desempeño de éstos puede variar en función del valor de k por lo que se buscará estudiar el punto óptimo de ésta.

- El método de **Kriging** puede no ser tan útil cuando el reparto de los datos es regular ya que éste es muy potente cuando la malla no es estrictamente regular y, además, puede ser muy complejo si los focos son muy heterogéneos. Aun así, se espera que éste sea uno de los que mejor resultados presente.

- El método de **IDW** también se encuentra beneficiado por el reparto regular de los datos pero vuelve a ser ineficiente con patrones no lineales complejos.

- El método de **Árboles de decisión** ofrecen una alta fiabilidad a la hora de modelar relaciones no lineales y manejar muchos datos. Sin embargo, pueden requerir una buena calibración y, además, su desempeño puede variar mucho en función de la dependencia geográfica de la variable. Por otro lado no depende directamente de la proximidad espacial por lo que pueden no ser adecuados cuando la correlación espacial es crítica.

- El método de **Bosques aleatorios** puede ofrecer resultados mejores que los árboles de decisión y puede suavizar las predicciones mostradas en el caso de árboles de decisión. A pesar de esto, puede producir sobre-ajustes o desequilibrios importantes.

5.1. Vecinos cercanos (NN) y vecinos naturales (kNN)

El primer método a aplicar se trata del método más sencillo, narrado en el apartado 3.4.1. Este método se encuentra beneficiado por la presencia de datos regulares pero puede no capturar bien las dependencias espaciales no lineales.

La forma de aplicar y posteriormente analizar el funcionamiento de este método será estudiando las predicciones que éste haría sobre datos ya conocidos teniendo en cuenta los vecinos próximos, este método también es llamado *Leave-One-Out Cross Validation (LOOCV)*. De esta forma se podrá obtener tanto el error cometido en cada punto (Eq. 68) como el error relativo que permitirá conocer en términos generales el desempeño del modelo (Eq. 69). Posteriormente, se calculará la media de estos errores, calculados anteriormente para cada uno de los puntos.

Otro forma de evaluar el método será también el tiempo que conlleva la ejecución de éste, se buscará un modelo que, además de funcionar correctamente y tener un grado de confiabilidad alto sea también eficiente. Para este fin, se tiene que tener en cuenta que estas pruebas se están aplicando únicamente a un tipo de contaminante en un año concreto, posteriormente, se deberá aplicar también al resto de los datos así que será necesario que el tiempo de ejecución no sea excesivo.

$$E = |v_p - v_r| \tag{68}$$

$$E_r = \frac{E}{v_r} \cdot 100 \tag{69}$$

donde, v_p será el valor predicho por el método y v_r será el valor real en ese punto.

El código del que se ha hecho uso para la aplicación del método se muestra en Lst. 9. Para este se ha hecho uso de la librería *FNN* que ofrece una versión optimizada y más eficiente del algoritmo kNN. Esta librería, además de permitir aplicar clasificación (como lo hace la librería *class*) permite aplicar regresión lo cual beneficia teniendo en cuenta nuestros datos.

La función principal a la hora de aplicar este método es *knn.reg()*. Esta función requiere de datos de entrenamiento, los cuales deben ser las variables explicativas sin la variable a predecir - en este caso concreto serán latitud y longitud sin tener en cuenta la concentración de contaminantes - por otro lado los datos de prueba, los

cuales serán las coordenadas del punto donde se quiere estimar la concentración; el vector con los valores reales de la variable objetivo correspondientes a cada punto de los datos de entrenamiento, es decir, el vector de concentraciones de contaminantes y, por último, el valor de vecinos que se tiene que buscar.

```
    tiempo <- system.time({
library(FNN)

# Numero de vecinos cercanos a evaluar
k <- 1
# Numero de valores a tener en cuenta en la prueba
n <- nrow(VV_CO)

# Creacion de variables vacias
predicciones <- numeric(length = nrow(VV_CO))
error <- numeric(length = nrow(VV_CO))
error_relativo <- numeric(length = nrow(VV_CO))

# Inicializacion de la variable de entrenamiento
X <- VV_CO[, c("Longitud", "Latitud")]


for (i in 1:n) {
  # El punto actual es el punto de prueba
  punto_test <- X[i, , drop = FALSE]

  # El resto de los puntos son los de entrenamiento
  X_train <- X[-i, , drop = FALSE]
  y_train <- VV_CO$CONCENTRACIN[-i]

  # Aplicar KNN
  prediccion <- knn.reg(train = X_train, test = punto_test, y = y_train
      , k = k)

  # Almacenar la prediccion
  predicciones[i] <- prediccion$pred

  # Calculo de errores
  error[i] <- abs(predicciones[i] - VV_CO$CONCENTRACIN[i])
  error_relativo[i] <- error[i] / VV_CO$CONCENTRACIN[i] * 100
}

```

```
data.frame(ID = 1:nrow(VV_CO), Real = VV_CO$CONCENTRACIN, Prediccion
    = predicciones, Error = error, ErrorRel = error_relativo)

# Calcular el promedio de error
promedio_error <- mean(error)
error_relativo_promedio <- mean(error_relativo, na.rm = TRUE)
})
```

Listing 9: Código para aplicación de los vecinos naturales con k=1.

Se debe tener en cuenta que, el algoritmo permite aplicar el método de k-vecinos más cercanos, es decir, el método permite aplicar al dato buscado el valor del vecino más cercano pero, además, permite aplicar el valor de la media de los "k"vecinos más cercanos. Teniendo en cuenta esto, en el código Lst. 9 se ha aplicado el método para $k = 1$ es decir, utilizando el modelo más sencillo dentro del enfoque de vecinos más cercanos. A partir de esta configuración inicial se han obtenido los primeros resultados. En la siguiente tabla (Tabla 5) se muestran los valores correspondientes a los 10 primeros datos evaluados.

ID	Real	Predicción	Error	Error Rel
1	0.215	0.227	1.200e-02	5.581395e+00
2	0.218	0.216	1.666e-03	7.645260e-01
3	0.214	0.220	6.333e-03	2.959502e+00
4	0.221	0.222	1.000e-03	4.524887e-01
5	0.220	0.221	6.666e-04	3.030303e-01
6	0.219	0.219	0.000e+00	0.000000e+00
7	0.217	0.227	1.000e-02	4.608295e+00
8	0.243	0.217	2.5667e-02	1.056241e+01
9	0.220	0.219	1.000e-03	4.545455e-01
10	0.223	0.220	2.667e-03	1.195815e+00

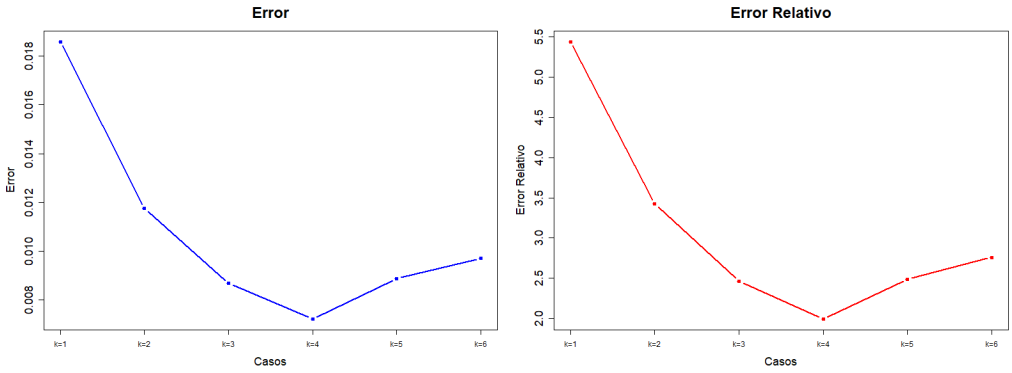
Tabla 5: Predicción y errores – Vecinos naturales ($k = 1$).

Y, con este mismo método, la media de los valores de los errores y el tiempo de ejecución se muestran en la Tabla 6.

Se han modificado los valores de "k"desde 1 hasta 6 para poder evaluar como evolucionan el error, el error relativo y el tiempo de ejecución para cada caso, de esta manera, además, se conseguirá evaluar el número de vecinos óptimo para poder estudiar la concentración de contaminantes en un punto de esta malla y con la aplicación de este método. Los resultados obtenidos se muestra en la Fig.54.

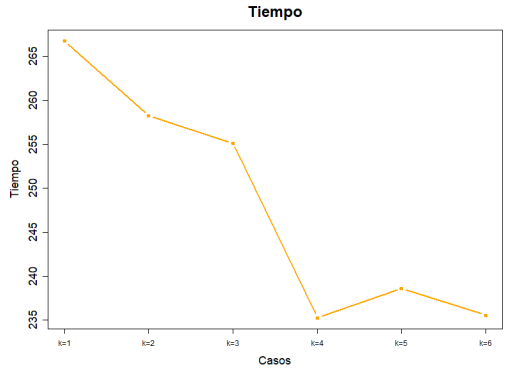
Tipo	Valor
Error promedio	0.018578
Error relativo [%]	5.437721
Porcentaje de confianza [%]	94.562279
Tiempo (*elapsed*) [s]	266.78

Tabla 6: Errores y tiempo de ejecución – Vecinos naturales ($k = 1$).



(a) Error promedio

(b) Error relativo



(c) Tiempo de ejecución

Figura 54: Evolución de los errores y el tiempo de ejecución con el número de vecinos cercanos "k". Referencias: elaboración propia

Como se puede observar en las Fig.54a y 54b el valor óptimo de vecinos que se van a evaluar es $k = 4$, a partir de este número se puede observar como el error vuelve a crecer. Por tanto, en $k = 4$ los valores de error y tiempo de ejecución se muestran en la Tabla 7.

Tipo	Valor
Error promedio	0.007231609
Error relativo [%]	1.996351
Porcentaje de confianza [%]	98.003649
Tiempo (*elapsed*) [s]	235.25

Tabla 7: Errores y tiempo de ejecución – Vecinos naturales ($k = 4$).

5.2. Kriging

El método de Kriging, como ya se ha comentado en apartados anteriores, resulta ser una técnica de interpolación espacial óptima para muestrear datos en ubicaciones donde no se cuenta con ellos.

El primer paso previo a la aplicación del método de Kriging será encontrar el modelo de variograma que mejor funcione para la estructura de los datos de los que se partía. En este caso se realizaron numerosas pruebas donde se dejaron a un lado aquellos que no conseguían una converger, o cuya forma no representaba de manera realista la variabilidad encontrada.

Entre los datos de monóxido de carbono se valoraron los modelos esférico y exponencial. Para ello son de utilidad las librerías *gstat* que permite aplicar el método de Kriging y crear variogramas y, además, la librería *sp* que permite convertir los puntos en un Spacial Data Frame (SPDF). El formato *SPDF* es el formato requerido por la librería *gstat* para los análisis geoestadísticos ya que éste es un formato espacial que define la geometría (posición) de cada punto.

El código en el que se modifica la estructura de los datos para pasarlos a formato *spdf* y, por otro lado, se aplican los dos tipos de ajuste mediante la función *fit.variogram* se muestra en Lst. 10.

```
98  # Seleccion de las columnas de coordenadas
    coords <- VV_CO[, c("Longitud", "Latitud")]
100 datos <- VV_CO["CONCENTRACIN"]

102 # Paso de datos a formato SPDF
    puntos <- SpatialPoints(coords)
104 spdf <- SpatialPointsDataFrame(puntos, data = datos)

106 # Creacion del variograma experimental
    variogram_exp <- variogram(CONCENTRACIN ~ 1, spdf)
```

```
variograma <- variogram(CONCENTRACIN ~ 1, spdf, width = 0.3)

# Aplicacion de los variogramas exponencial y esferico respectivamente
ajuste_exp <- fit.variogram(variograma, model = vgm(psill = 60, "Exp",
    range = 200, nugget = 1))
ajuste_esf <- fit.variogram(variograma, model = vgm(psill = 60, "Sph",
    range = 200, nugget = 1))

# Creacion de im genes y descarga de estas
png("Directorio de descarga", width = 8, height = 4, units = "in", res
    = 300)
plot(variograma, ajuste_exp, main = "Ajuste Exponencial CO")
dev.off()
png("Directorio de descarga", width = 8, height = 4, units = "in", res
    = 300)
plot(variograma, ajuste_esf, main = "Ajuste Esf rico CO")
dev.off()
```

Listing 10: Código para la creación de variogramas exponencial y esférico.

Los resultados de aplicación de los dos variogramas se muestran en las Fig.55.

Para analizar el ajuste, además de gráficamente, se ha valorado a través del coeficiente de determinación (R^2). El método aplicado es útil para evaluar de forma adaptada la calidad del ajuste del variograma.

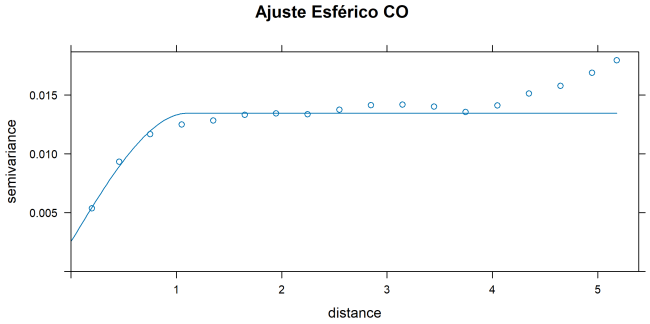
Este método se calcula haciendo uso de la Eq. 70.

$$R^2 = 1 - \frac{SSE}{SST} \tag{70}$$

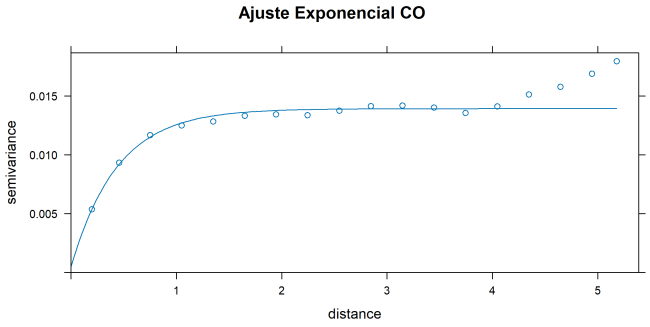
donde:

- SSE es el error cuadrático del modelo ajustado. Es decir, la suma de los cuadrados de los errores.
- SST es la suma de todos los cuadrados. Es decir, la variabilidad total respecto a la media del variograma.

El código para aplicar este coeficiente en nuestra base de datos se muestra en Lst. 11.



(a) Variograma con ajuste esférico



(b) Variograma con ajuste exponencial

Figura 55: Variogramas con diferentes ajustes para la variable CO. Referencias: elaboración propia

```
122 # Coeficiente de determinacion para el ajuste esferico
    SSErr<-attr(ajuste_esf,"SSErr")
124 weig<-variograma$np / variograma$dist^2
    SStot<- sum(weig*(variograma$gamma - mean(variograma$gamma))^2)
126 (R2<-1-SSErr / SStot)

128 # Coeficiente de determinacion para el ajuste exponencial
    SSErr<-attr(ajuste_exp,"SSErr")
130 weig<-variograma$np / variograma$dist^2
    SStot<- sum(weig*(variograma$gamma - mean(variograma$gamma))^2)
132 (R2<-1-SSErr / SStot)
```

Listing 11: Código para la determinación del coeficiente de correlación.

Los resultados para los coeficientes de correlación son:

- Coeficiente de correlación esférico: 0.9871946

- Coeficiente de correlación exponencial: 0.9935092

Se puede determinar, tanto gráficamente como mediante el coeficiente de correlación, que el ajuste que mejor funciona para la varianza de CO es el ajuste exponencial.

A partir de este momento se podrá aplicar el método de Kriging. Para ello el primer paso será definir la cuadrícula en la que se interpolarán los datos, cuanto más grande sea ésta más tiempo de estimación se necesitará. En esta prueba se ha hecho uso de 10.000 puntos en la cuadrícula, los cuales, son menos que los que teníamos inicialmente (23.000). Esto quiere decir que la aplicación de este método con ese tamaño de cuadrícula no sería de especial interés.

Posterior a la definición de la cuadrícula se ha aplicado el método de Kriging haciendo uso de la función *gstat::krige*. La aplicación del código se muestra en Lst. 12.

```
tiempo <- system.time({

# Generacion de la malla de tama o 100 x 100
grd <- expand.grid(Longitud = seq(min(VV_CO$Longitud), max(VV_CO$
    Longitud), length.out = 100),
                   Latitud = seq(min(VV_CO$Latitud), max(VV_CO$Latitud)
                       , length.out = 100))
# Definicion de las coordenadas de la cuadr cula
coordinates(grd) <- ~Longitud + Latitud

# Interpolacion con el modelo ajustado (exponencial)
ns.k.exp <- gstat::krige(CONCENTRACIN ~ 1, spdf, grd, model = ajuste_
    exp)
interpolacion <- ns.k.exp
})
```

Listing 12: Código para la aplicación del método de Kriging.

El código ha sido adaptado de la versión actual de R a partir del libro *Métodos de interpolación espacial para el mapeo de la riqueza de especies usando R* [75], con el apoyo de herramientas de inteligencia artificial como ChatGPT (OpenAI).

Posterior a la aplicación del método de Kriging se procede ha realizar el cálculo del error con el fin de, posteriormente, comparar este método con los anteriores. A este fin, se ha vuelto a calcular el error promedio, error relativo y el porcentaje de

confianza haciendo uso de una función contenida en la librería *gstat*, denominada *gstat::krige.cv*. Esta función permite realizar una validación cruzada y así poder estimar correctamente el ajuste y el error. En la Tabla 8 se muestran los valores obtenidos tras la ejecución del código Lst. 13.

```
set.seed(123)
spdf_sample <- spdf[sample(1:nrow(spdf), 1000), ]
cv_sample <- gstat::krige.cv(CONCENTRACION ~ 1, spdf_sample, model =
    ajuste_exp, nfold = 1000)

error_promedio <- abs(mean(cv_sample$residual))
error_relativo <- mean(abs(cv_sample$residual / cv_sample$observed), na
    .rm = TRUE) * 100  # En porcentaje

lower <- cv_sample$var1.pred - 1.96 * sqrt(cv_sample$var1.var)
upper <- cv_sample$var1.pred + 1.96 * sqrt(cv_sample$var1.var)

# Contar cuntos valores reales est n dentro del intervalo
dentro_intervalo <- cv_sample$observed >= lower & cv_sample$observed <=
     upper

# Porcentaje de confianza
porcentaje_confianza <- mean(dentro_intervalo, na.rm = TRUE) * 100

# Porcentaje de confianza
porcentaje_confianza <- mean(dentro_intervalo, na.rm = TRUE) * 100
```

Listing 13: Cálculo de error en el método de Kriging.

Para el desarrollo del código Lst. 13 se ha realizado un ensayo sobre una muestra de 1000 datos, ya que ejecutar el código para todos los datos de la variable podría tomar varias horas. Se entiende así que estos datos darán un valor de error representativo.

Tipo	**Valor**
Error promedio	0.000699
Error relativo [%]	8.060691
Porcentaje de confianza [%]	91.93
Tiempo (*elapsed*) [s]	8452.60

Tabla 8: Errores y tiempo de ejecución – Kriging.

Es importante recalcar que el tiempo de ejecución ha sido de 2 horas y 21 mi-

nutos. A pesar del costo computacional, el resultado ha sido una malla con menos puntos de los que partíamos, es decir, hemos pasado de unos 23.000 puntos a unos 10.000. Además, el error relativo pese a no ser excesivamente alto tampoco indica un ajuste de los datos casi perfecto. Todas estas variables serán importantes para las conclusiones finales.

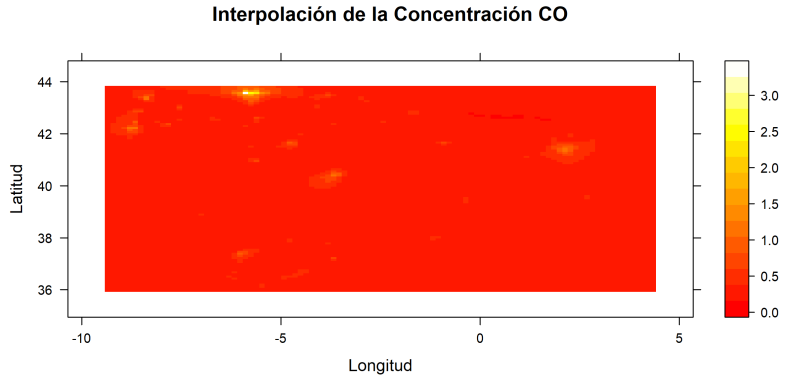
Por otro lado, el porcentaje de de confianza obtenido no se trata del mismo del apartado anterior (basado en el Error relativo) sino uno más robusto. Este porcentaje de confianza indica que, el 97.5 [%] de las ocasiones los datos cayeron dentro del intervalo de confianza del 95 [%] lo cual indica, en general, un buen intervalo. En caso de calcularlo basado en el Error Relativo el resultado sería 91,93[%].

Para obtener una visión más completa de en qué ha resultado el método de Kriging se ha realizado un gráfico que muestra en diferentes colores los niveles de concentración de CO tras la interpolación. El resultado de esta gráfica se muestra en la Fig.56.

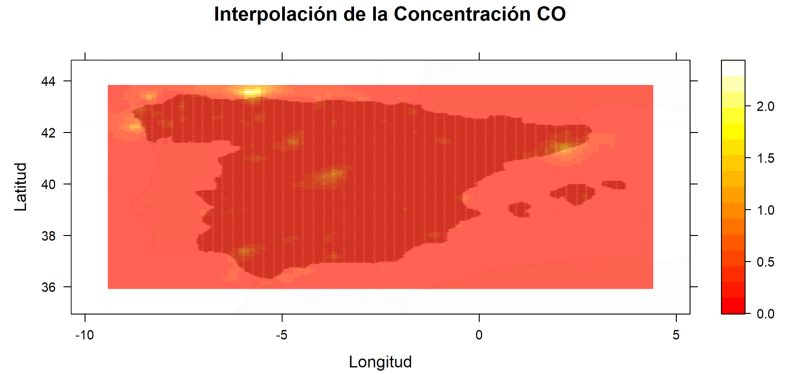
Analizando la Fig.56b se puede observar numerosos puntos de alta concentración de contaminante CO. Estos se localizan en zonas cercanas a Madrid, Barcelona, Sevilla, Cantabria o Asturias y partes de Galicia.

Estas imágenes se han obtenido aplicando el código Lst. 14.

```
164
library(sp)
166 library(lattice)

168 spplot(interpolacion, "var1.pred", main = "Interpolaci n de la
    Concentraci n CO",
        col.regions = heat.colors(100),
170     at = seq(min(interpolacion$var1.pred), max(interpolacion$var1.
            pred), length.out = 100))

172 png("Directorio del archivo", width = 8, height = 4, units = "in", res
    = 300)

174 levelplot(var1.pred ~ Longitud + Latitud, data = as.data.frame(
    interpolacion),
            main = "Interpolaci n de la Concentraci n CO",
176         col.regions = heat.colors(100),
            xlim = c(min(VV_CO$Longitud) - 1, max(VV_CO$Longitud) + 1),
               # Aumentar el rango de Longitud
```



(a)



(b) Superposición con el mapa de España

Figura 56: Concentraciones de CO tras la interpolación mediante Kriging. Referencias: elaboración propia

```
178            ylim = c(min(VV_CO$Latitud) - 1, max(VV_CO$Latitud) + 1))
dev.off()
```

Listing 14: Obtención de imágenes del método de Kriging.

5.3. IDW

El siguiente método a tener en cuenta se encuentra comentado en el apartado 3.4.4. Este método está basado en la distancia Euclídea que, a diferencia de los vecinos cercanos, da peso a la distancia desde tu punto de estudio al resto de los puntos. Este método también se trata de uno de los más aceptados y sencillos, y en principio, más robusto.

La evaluación del método se volverá a realizar mediante validación cruzada (*Leave-One-Out Cross Validation*) es decir, interpolando los datos en un punto ya conocido y comparándolo con la solución real en el mismo.

Con la ayuda de la herramienta *Chat GPT* se ha elaborado el código para la ejecución de este método y éste se muestra en Lst. 15.

```
    tiempo <- system.time({
# Creacion de vectores para el almacenamiento de resultados
errores_absolutos <- numeric(nrow(VV_CO))
errores_relativos <- numeric(nrow(VV_CO))
confianzas <- numeric(nrow(VV_CO))

for (i in 1:nrow(VV_CO)) {

  # Punto a predecir
  punto_real <- VV_CO[i, ]
  localizacion <- c(punto_real$Latitud, punto_real$Longitud)

  # Datos restantes (excluyendo el punto a predecir)
  CO_restantes <- VV_CO[-i, ]

  # Calculo de distancias al resto de puntos
  distancias <- sqrt((CO_restantes$Latitud - localizacion[1])^2 +
                     (CO_restantes$Longitud - localizacion[2])^2)
  distancias[distancias == 0] <- 1e-10

  # Seleccion de los k puntos m s cercanos
  k <- 4
  puntos_cercanos <- CO_restantes[order(distancias), ][1:k, ]
  distancias_k <- distancias[order(distancias)][1:k]

  # Pesos
  pesos <- 1 / (distancias_k)
  pesos_normalizados <- pesos / sum(pesos)

  # Estimacion de la concentracion
  concentraciones <- puntos_cercanos$CONCENTRACIN
  conc_estim <- sum(pesos_normalizados * concentraciones)

  # Calculo de errores
  error_abs <- abs(conc_estim - punto_real$CONCENTRACIN)
  error_rel <- error_abs / punto_real$CONCENTRACIN
```

```
  confianza <- (1 - error_rel) * 100
  errores_absolutos[i] <- error_abs
  errores_relativos[i] <- error_rel
  confianzas[i] <- confianza
}

error_medio_absoluto <- mean(errores_absolutos)
error_medio_relativo <- mean(errores_relativos)
})
```

Listing 15: Evaluación del método IDW.

Para el caso de k=4 (el estudio del punto óptimo se ha calculado en el subapartado 5.4). Los resultados de error absoluto, relativo y confianza se muestran en la Tab.9

Tipo	Valor
Error promedio	0.0072
Error relativo [%]	1.99
Porcentaje de confianza [%]	98.01
Tiempo (*elapsed*) [s]	163.23

Tabla 9: Errores y tiempo de ejecución – IDW

5.4. IDW con potencia de 2

Este método dará aun más importancia a la distancia ya que esta se encontrará elevada al cuadrado. Se trata del mismo código anterior pero modificando la línea de código:

```
    pesos <- 1 / (distancias_k)
```

Listing 16: Código para calcular IDW simple.

De la siguiente manera:

```
    pesos <- 1 / (distancias_k^2)
```

Listing 17: Código para calcular IDW simple.

Es importante recalcar que, al igual que en el caso de los vecinos naturales también se puede seleccionar el número de puntos que se tendrá en cuenta para la interpolación. En este primer caso se ha realizado con 4 puntos y el resultado se muestra en la Tab.10.

Tipo	Valor
Error promedio	0.0072
Error relativo [%]	1.98
Porcentaje de confianza [%]	98.02
Tiempo (*elapsed*) [s]	163.23

Tabla 10: Errores y tiempo de ejecución – IDW^2 (k=4).

Tras el análisis de esta primera prueba se puede concluir que los resultados obtenidos son considerablemente buenos ya que el error relativo es muy bajo y, por tanto, el porcentaje de confianza alto. Por otro lado, el tiempo de ejecución del programa parece razonable teniendo en cuenta el número de datos que se poseían.

Finalmente se ha optado por determinar, además del error, el número de puntos óptimo a considerar. Para poder analizar gráficamente este punto se han representado los valores en la Fig.57.

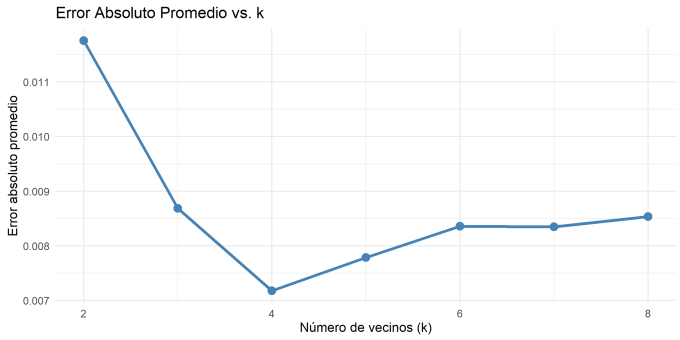


Figura 57: Error promedio en función del valor k para el método IDW^2. Referencias: elaboración propia

Para graficar estos valores se ha hecho uso del código mostrado en Lst. 18.

```
228 library(ggplot2)
    png("Directorio del archivo", width = 8, height = 4, units = "in", res
        = 300)
230 grafico_error_abs <- ggplot(resultados_k, aes(x = k, y = error_absoluto
        _promedio)) +
      geom_line(color = "steelblue", size = 1.2) +
232   geom_point(color = "steelblue", size = 3) +
      theme_minimal() +
234   labs(title = "Error Absoluto Promedio vs. k",
           x = "N mero de vecinos (k)",
236        y = "Error absoluto promedio")
    print(grafico_error_abs)
```

```
238 dev.off()
```

Listing 18: Código para graficar el error promedio.

Como se puede observar, el punto de error mínimo es para k=4, es decir, se considerará como resultado de error para la aplicación de este método los mostrados en la Tab.10.

5.5. IDW con diversas potencias

Debido a que la potencia a la que se somete la distancia controla el grado de influencia que tienen los puntos se entiende que a mayor potencia mayor peso tendrán los puntos más próximos frente a los lejanos.

En este apartado se analizará el efecto de variar la potencia de la distancia en el rendimiento del modelo. Manteniendo fijo el valor de k=4 se evaluarán potencias entre 2 y 8, comparando así los errores de predicción.

El código desarrollado para estudiar este caso se muestra en Lst. 19.

```
240 # Creacion del data frame
    resultados_potencia <- data.frame(
242   potencia = numeric(),
      error_absoluto_promedio = numeric(),
244   error_relativo_promedio = numeric(),
      confianza_promedio = numeric()
246 )

248 # Valores de potencia a evaluar
    potencias <- 2:16
250 k <- 4

252 for (p in potencias) {

254   errores_absolutos <- numeric(nrow(VV_CO))
      errores_relativos <- numeric(nrow(VV_CO))
256   confianzas <- numeric(nrow(VV_CO))

258   for (i in 1:nrow(VV_CO)) {

260     punto_real <- VV_CO[i, ]
        localizacion <- c(punto_real$Latitud, punto_real$Longitud)
```

```
262     CO_restantes <- VV_CO[-i, ]

264     distancias <- sqrt((CO_restantes$Latitud - localizacion[1])^2 +
                           (CO_restantes$Longitud - localizacion[2])^2)
266     distancias[distancias == 0] <- 1e-10

268     puntos_cercanos <- CO_restantes[order(distancias), ][1:k, ]
        distancias_k <- distancias[order(distancias)][1:k]
270
        pesos <- 1 / (distancias_k^p)
272     pesos_normalizados <- pesos / sum(pesos)

274     concentraciones <- puntos_cercanos$CONCENTRACIN
        conc_estim <- sum(pesos_normalizados * concentraciones)
276
        error_abs <- abs(conc_estim - punto_real$CONCENTRACIN)
278     error_rel <- error_abs / punto_real$CONCENTRACIN
        confianza <- (1 - error_rel) * 100
280
        errores_absolutos[i] <- error_abs
282     errores_relativos[i] <- error_rel
        confianzas[i] <- confianza
284   }

286   error_medio_absoluto <- mean(errores_absolutos)
      error_medio_relativo <- mean(errores_relativos)
288   confianza_promedio <- mean(confianzas)

290   resultados_potencia <- rbind(resultados_potencia, data.frame(
        potencia = p,
292     error_absoluto_promedio = error_medio_absoluto,
        error_relativo_promedio = error_medio_relativo,
294     confianza_promedio = confianza_promedio
      ))
296 }
```

Listing 19: Cálculo de errores en función de la potencia de la distancia.

Los resultados obtenidos de este estudio se muestran en la Fig.58.

Como se puede observar en la gráfica la exactitud aumenta (el error disminuye) en cuanto la potencia es más alta. Es decir, el error disminuye cuanta más importancia tiene la distancia en la estimación.

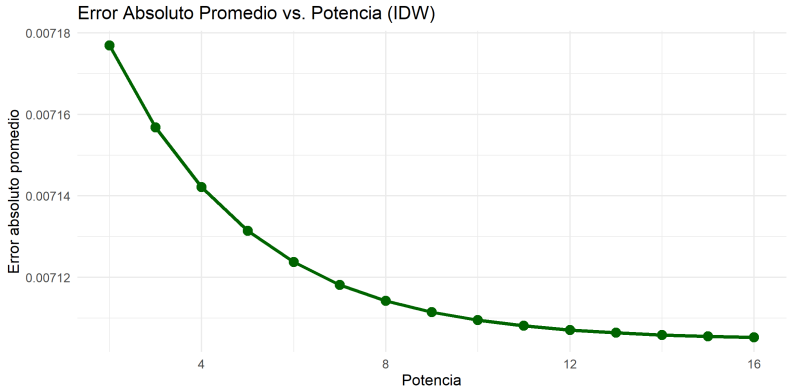


Figura 58: Evolución del error absoluto en función de la potencia. Referencias: elaboración propia

Esto, a pesar de tener un carácter positivo también puede tener algunas consecuencias negativas como por ejemplo:

- El modelo se volverá más sensible a valores cercanos. Si el modelo presenta un valor atípico, la predicción se verá muy afectada.
- Se puede dar lugar a sobre-ajustes locales lo que puede empeorar la generalización del modelo.
- En estudios geoespaciales potencias muy altas pueden generar superficies interpoladas poco suaves, con contornos abruptos y poco realistas, lo que puede ir en contra de la continuidad espacial de algunos fenómenos naturales.

A modo de ejemplo se adjunta la Tab.11 en el que, a modo de resumen, se analizan los resultados de algunas de las potencias calculadas.

Tipo	**p=4**	**p=6**	**p=8**	**p=12**
Error promedio	0.007142	0.007123	0.007114	0.007107
Error relativo [%]	1.974340	1.969792	1.967537	1.965901
Porcentaje de confianza [%]	98.02566	98.03021	98.03246	98.03410

Tabla 11: Errores y tiempo de ejecución – IDW con diferentes potencias (p)

Hay que tener en cuenta que el modelo debe ser replicable al resto de contaminantes ya que actualmente se está haciendo uso únicamente de la base de datos de CO para el año 2019. Teniendo en cuenta que la variación entre los errores es del orden de las centomilésimas ($7,17 \times 10^{-5}$) elegir una potencia menor y más estandarizada no va a suponer grandes cambios en la solución final.

5.6. Árboles de decisión

Se ha hablado anteriormente de los árboles de decisión (apartado 3.1.3) como modelo para la posterior predicción de posibles enfermedades asociadas con la contaminación. Por ese mismo motivo, se ha valorado la posibilidad de hacer uso de esta herramienta también para la interpolación y de esta manera valorar su funcionamiento.

Los árboles de decisión no son una técnica habitual para la interpolación espacial. Para este trabajo el modelo dividirá el espacio en regiones rectangulares según las coordenadas (por ejemplo, *si Latitud < 42.3 y Longitud > -1.5...*) creando así una rejilla adaptativa, esto simplificará el espacio y otorgará dinamismo y velocidad al modelo.

Sin embargo, este modelo también enfrenta contra-indicaciones o limitaciones. Esto se debe a que no modela continuidad espacial como lo hacen Kriging o IDW y, por tanto, puede generar saltos abruptos no realistas.

El código para la aplicación del modelo de árboles de decisión se muestra en Lst. 20. Para el desarrollo de éste se ha hecho uso de la herramienta *Chat GPT* y modificaciones necesarias para su correcto funcionamiento. Se ha implementado la librería requerida para hacer uso de árboles de decisión en R, *rpart*.

```
if (!require(rpart)) install.packages("rpart")
298 library(rpart)

300 tiempo <- system.time({
# Creacion de vectores con el fin de almacenar errores
302 errores_absolutos <- numeric(nrow(VV_CO))
errores_relativos <- numeric(nrow(VV_CO))
304
# Validacion cruzada Leave-One-Out
306 for (i in 1:nrow(VV_CO)) {

308   # Punto a predecir
  punto_real <- VV_CO[i, ]
310
  # Conjunto de entrenamiento
312   datos_entrenamiento <- VV_CO[-i, ]

314   # Entrenamiento del modelo
```

```
  modelo_cv <- rpart(CONCENTRACIN ~ Longitud + Latitud, data = datos_
      entrenamiento, method = "anova")
316  prediccion <- predict(modelo_cv, newdata = punto_real)

318  # Calculo de errores
  error_abs <- abs(prediccion - punto_real$CONCENTRACIN)
320  error_rel <- error_abs / punto_real$CONCENTRACIN

322  errores_absolutos[i] <- error_abs
  errores_relativos[i] <- error_rel
324 }

326 error_medio_absoluto <- mean(errores_absolutos)
error_medio_relativo <- mean(errores_relativos)
328 })
```

Listing 20: Código para la generación y evaluación de árboles aleatorios como modelo de interpolación.

Como se puede observar en el resumen de la Tab.12 los resultados obtenidos mediante árboles de decisión presentan un error relativo considerablemente mayor a otros métodos de interpolación.

Tipo	Valor
Error promedio	0.045494
Error relativo [%]	16.45
Porcentaje de confianza [%]	83.55
Tiempo (*elapsed*) [s]	1743.59

Tabla 12: Errores y tiempo de ejecución – Árboles de decisión.

En general, estos resultados permiten anticipar que los árboles de decisión podrían no ser la opción óptima para este tipo de tareas o, al menos, con este tipo de nube de puntos.

5.7. Bosques aleatorios

Los bosques aleatorios se tratan de una extensión de los árboles de decisión. El resultado de estos es una combinación de árboles que buscan mejorar la precisión y reducir el sobreajuste.

Haciendo uso de este método se entrenan varios árboles de decisión de manera simultánea sobre un subconjunto aleatorio de los datos. En cada nodo de los árboles, se elegirá aleatoriamente un subconjunto de características para la toma de decisiones. El método realizará una clasificación en que se indica qué árboles se van a seleccionar. En el programa R se hace uso de la librería *Random Forest* para su aplicación

Por otro lado, éste se trata de un modelo mucho más complejo que los árboles de decisión y, aunque es cierto que se mejora la precisión, también supone un mayor costo computacional. Debido a esto, resultó inviable aplicar el método de validación cruzada *Leave-One-Out* ya que el tiempo de ejecución sería en este caso excesivo.

De esta manera, se tomó la decisión de utilizar el método *10-fold-cross-validation* en el cual se entrenan 10 modelos usando el 90 % de los datos ahorrando así tiempo. Para la aplicación de este método de comprobación sería necesario hacer uso de la librería *caret*.

El código desarrollado para la aplicación de este método se encuentra en Lst. 21.

```
    # Cargar librer as necesarias
330 if (!require(caret)) install.packages("caret")
    if (!require(randomForest)) install.packages("randomForest")
332 library(caret)
    library(randomForest)
334
    tiempo <- system.time({
336 # Configurar validaci n cruzada 10-fold
    set.seed(123)
338 control <- trainControl(method = "cv", number = 10, savePredictions = "
    final")

340 # Entrenar el modelo con caret
    modelo_caret_rf <- train(CONCENTRACIN ~ Longitud + Latitud,
342                          data = VV_CO,
                             method = "rf",
344                          trControl = control,
                             ntree = 100)
346
    # Mostrar resumen de rendimiento de caret
348 print(modelo_caret_rf)

350 # Obtener predicciones cruzadas y valores reales
```

```
predicciones <- modelo_caret_rf$pred$pred
observados <- modelo_caret_rf$pred$obs

# Calcular errores
errores_absolutos <- abs(predicciones - observados)
errores_relativos <- errores_absolutos / observados

# M tricas finales
error_medio_absoluto <- mean(errores_absolutos)
error_medio_relativo <- mean(errores_relativos) * 100  # en porcentaje

# Mostrar resultados
cat("Error absoluto promedio:", round(error_medio_absoluto, 6), "\n")
cat("Error relativo promedio:", round(error_medio_relativo, 2), "%\n")
})
```

Listing 21: Código para la aplicación de bosques aleatorios.

Los resultados obtenidos de la aplicación de este método se muestran a modo de resumen en la Tab.13.

Tipo	Valor
Error promedio	0.009305
Error relativo [%]	2.64
Porcentaje de confianza [%]	97.36
Tiempo (*elapsed*) [s]	441.17
Tiempo *Leave-One-Out* [s]	1013691

Tabla 13: Errores y tiempo de ejecución – Bosques aleatorios.

A pesar que el tiempo de ejecución del programa en realidad fue de 441 [s], para poder comparar el modelo con los otros modelos ensayados con el método *Leave-One-Out* se ha estimado el tiempo que se tardaría aplicando éste. Para ello hay que tener en cuenta que el nuevo método ha entrenado únicamente 10 modelos frente a los 23.000 que hubiera entrenado el modelo anterior. En conclusión, aplicando el método de *Leave-One-Out* hubieran sido necesarias aproximadamente 280 [h!].

5.8. Conclusiones

Una vez haber evaluado numerosos métodos de interpolación al mismo conjunto de datos distribuidos en una cuadrícula regular y con un comportamiento no lineal.

Con el objetivo de determinar cuál de ellos ofrece un mejor equilibrio entre precisión, fiabilidad y eficiencia, se han comparado las métricas de error relativo, nivel de confianza y tiempo de cálculo. Los resultados obtenidos en la Tab.14 permitirán identificar las técnicas más adecuadas para este tipo de escenario.

Modelo	**Error relativo** [%]	**Confianza** [%]	**Tiempo** [s]
NN [1](k=1)	5.437721	94.562279	266.78
kNN [2](k=4)	1.996351	98.003649	235.25
Kriging	8.060691	97.500000	8452.60
IDW	1.990000	98.010000	163.23
IDW^2	1.980000	98.020000	163.23
IDW^6	1.969792	98.030210	s.d.[3]
Árboles de decisión	16.450000	83.550000	1743.59
Bosques aleatorios	2.640000	97.360000	1013691.00

Tabla 14: Comparativa de métodos de interpolación y errores asociados.

Los resultados obtenidos muestran que los métodos como los árboles de decisión, bosques aleatorios y Kriging presentan errores elevados respecto a otros métodos más sencillos. Además, el coste computacional de cualquiera de estos tres resulta excesivo para el resultado obtenido. Por tanto, se ha optado por no escoger ninguno de estos métodos para el resultado final.

Por otro lado el método de vecinos cercanos o naturales (NN y kNN) presentan un desempeño algo más valioso. El nivel de confianza es alto en ambos casos aunque algo mayor en el caso de vecinos naturales. Ésto convierte al método kNN es una alternativa válida que supone no tan alto coste computacional.

Finalmente, los resultados obtenidos por el método de interpolación inversa de la distancia (IDW), especialmente con potencias IDW^2 e IDW^6, ofrecen un muy correcto rendimiento global con errores relativos inferiores al 2 % y costes computacionales muy reducidos. Estos enfoques destacan como los más adecuados para datos no lineales distribuidos en cuadrícula lineal ya que consiguen capturar adecuadamente las variaciones locales siendo además un método sencillo sin excesivo coste computacional.

En conclusión, tras analizar el rendimiento de los distintos métodos de interpolación sobre un conjunto de datos distribuidos regularmente y con comportamientos

[1]NN: vecinos cercanos
[2]kNN: vecinos naturales
[3]s.d.: sin dato disponible.

no lineales, se constata que las soluciones más simples y locales, como el método de kNN o, especialmente el método IDW con potencias, ofrecen un equilibrio óptimo entre precisión y eficiencia. Por tanto, se hará uso del método de IDW^6 para la aplicación de la interpolación al conjunto de datos a estudiar.

5.9. Aplicación de la interpolación a la base de datos

El siguiente paso del proceso será la aplicación de interpolación a la base de datos de parámetros de salud (*dat.salud*). Se buscará asociar cada fila de esta base de datos a una concentración de CO, NO_2, $PM_{2,5}$, PM_{10}, O_{3_1}, $O_{3_{26}}$ y SO_2. También se debe tener en cuenta que para cada uno de estos contaminantes se tienen datos desde 2008 a 2019 y, debido a que no se tienen datos que hagan referencia a fechas en la base de datos, se calculará la concentración de cada contaminante en cada uno de estos años.

El proceso de aplicación del método IDW descrito en Lst. 15 presenta limitaciones significativas en lo referente al tiempo de ejecución. El código para la aplicación de este mismo método para cada uno de los individuos de la base de datos de parámetros de salud (*dat.salud*) requiere un tiempo promedio de 4483.74 [s] (equivalente a unas 1,24 [h!]) por cada base de datos correspondiente a cada uno de los contaminantes. Considerando que la interpolación debe aplicarse a siete contaminantes a lo largo de doce años, el tiempo total estimado de ejecución para completar todo el proceso asciende a aproximadamente 104,16 [h!]. El código de aplicación de este método para la base de datos se muestra en Lst. 22.

```
366 tiempo <- system.time({

368   predicciones <- numeric(nrow(dat.salud))
for (i in 1:nrow(dat.salud)) {
370   # Punto a predecir
   punto_real <- dat.salud[i, ]
372   punto_real <- c(punto_real$Latitud, punto_real$Longitud)

374   # Calcular distancias a todos los otros puntos
   distancias <- sqrt((`2019_CO`$Latitud - punto_real[1])^2 +
376                      (`2019_CO`$Longitud - punto_real[2])^2)

378   distancias[distancias == 0] <- 1e-10
```

```
380   k <- 4
      puntos_cercanos <- `2019_CO`[order(distancias), ][1:k, ]
382   distancias_k <- distancias[order(distancias)][1:k]

384   pesos <- 1 / (distancias_k)
      pesos_normalizados <- pesos / sum(pesos)
386
     # Estimar concentraci n
388   concentraciones <- puntos_cercanos$CONCENTRACIN
      conc_estim <- sum(pesos_normalizados * concentraciones)
390
      predicciones[i] <- conc_estim
392 }

394 predicciones
    })
```

Listing 22: Aplicación de la interpolación por el método de IDW a cada individuo

La ineficiencia se debe principalmente al enfoque algorítmico empleado. En concreto:

- Ineficiencia en el manejo del bucle debido a que R no está optimizado para ejecutar bucles extensos de manera eficiente.
- En cada iteración se recalculan las distancias entre el punto actual y todos los puntos de referencia desde cero.
- La función *order* tiene una complejidad computacional alta y, por tanto, ejecutar ésta en cada una de las iteraciones del bucle supone un tiempo.
- Ausencia de métodos de búsqueda eficientes.

Con la ayuda de la herramienta *ChatGPT* de *OpenAI* se ha buscado un método que pueda realizar exactamente el mismo proceso de forma eficiente. La solución ha sido emplear el paquete *FNN* de R (utilizada también en la aplicación del método kNN), que implementa algoritmos avanzados de búsqueda de vecinos cercanos evitando así el cálculo repetitivo de distancias para cada punto mediante estructuras de datos especializadas.

La función utilizada a este fin ha sido *get.knnx*. Esta función construye una estrutura de datos tridimensional, específicamente un árbol k-d. Por cada punto de

consulta la función devuelve los índices de los vecinos cercanos y las distancias correspondientes a estos vecinos. Con la aplicación de este método se podrían presentar alguna diferencias con respecto a las predicciones de Lst. 22 debido a que la función *order* desempata por orden de aparición posibles puntos que se encuentren a la misma distancia mientras que la función *get.knnx* hace uso de otros métodos. La aplicación de este segundo método se muestra en Lst. 23.

```
396 library(FNN)
    tiempo <- system.time({
398
    # Inicializaci n de valores
400 coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])
    coords_CO <- as.matrix(`2019_CO`[, c("Latitud", "Longitud")])
402
    k <- 4
404
    # Aplicaci n de la funci n de b squeda
406 vecinos <- get.knnx(data = coords_CO, query = coords_salud, k = k)

408 # Obtener los datos de distancias e  ndices  esperadas
    distancias <- vecinos$nn.dist
410 indices <- vecinos$nn.index

412 # Aplicaci n del m todo de IDW de la misma manera que anteriormente.
    distancias[distancias == 0] <- 1e-10
414 concentraciones_CO <- `2019_CO`$CONCENTRACIN
    pesos <- 1 / distancias
416 pesos_normalizados <- pesos / rowSums(pesos)
    concentraciones_vecinos <- matrix(concentraciones_CO[indices], ncol = k
        )
418
    # Predicci n final
420 predicciones_f <- rowSums(pesos_normalizados * concentraciones_vecinos)
    })
```

Listing 23: Búsqueda de vecinos cercanos para el método de IDW con la librería FNN

Con el fin de poder comparar si había o no desviaciones en las predicciones de ambos métodos se ha calculado la diferencia entre ambas (Lst. 24).

```
422 # Diferencia absoluta
    max(abs(predicciones - predicciones_f))
424
```

```
# Diferencia relativa
mean(abs(predicciones - predicciones_f) / predicciones_f)
```

Listing 24: Diferencia entre ambas predicciones.

El resultado de esta diferencia ha sido igual a cero, es decir, ambos métodos han dado como resultado exactamente las mismas predicciones. El tiempo de ejecución de este segundo método ha sido de 0.55 [s], es decir, menos de un segundo.

Gracias a esta mejora significativa en eficiencia, es factible aplicar este método para estimar las concentraciones de distintos contaminantes a lo largo de varios años. Se estima que el tiempo total de ejecución para el procesamiento completo será aproximadamente de 46,2 [s], permitiendo un análisis rápido y escalable.

Para la aplicación del código que permitiría interpolar los datos para cada uno de los contaminantes en cada uno de los escenarios se han aplicado al código Lst. 23 dos bucles que recorrerán cada uno de los contaminantes denominados con formato "*AAAA_CC*", donde .ªAAA"hace referencia al año y ÇC.ªl contaminante. El código en el que se aplica este método para todas las bases de datos se muestra en Lst. 25. El tiempo de ejecución de este programa ha sido de 61.19 [s], es decir, el tiempo esperado.

```
library(FNN)

dat.salud <- dat.salud[complete.cases(dat.salud[, c("Latitud", "
    Longitud")]), ]
coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])

tiempo <- system.time({
# Definicion de parametros
anios <- 2008:2019
contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")
k <- 4

coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])

# Inicializacion de la tabla de resultados
predicciones_total_idw <- data.frame(ID = seq_len(nrow(coords_salud)))

# Bucle para recorrer bases de datos
for (anio in anios) {
  for (cont in contaminantes) {
```

```
    nombre_var <- paste0(anio, "_", cont)

    if (exists(nombre_var)) {
      base <- get(nombre_var)

      if (all(c("Latitud", "Longitud", "CONCENTRACIN") %in% colnames(
          base))) {
        coords_contaminante <- as.matrix(base[, c("Latitud", "Longitud"
            )])
        concentraciones <- base$CONCENTRACIN

        # Vecinos mas cercanos
        vecinos <- get.knnx(data = coords_contaminante, query = coords_
            salud, k = k)
        distancias <- vecinos$nn.dist
        indices <- vecinos$nn.index

        distancias[distancias == 0] <- 1e-10

        # Metodo IDW
        pesos <- 1 / distancias
        pesos_normalizados <- pesos / rowSums(pesos)
        concentraciones_vecinos <- matrix(concentraciones[indices],
            ncol = k)

        # Predicciones
        predicciones <- rowSums(pesos_normalizados * concentraciones_
            vecinos)
        predicciones_total_idw[[nombre_var]] <- predicciones
      } else {
        warning(paste("Faltan columnas en", nombre_var))
      }
    } else {
      warning(paste("No existe la base de datos:", nombre_var))
    }
  }
}
})
```

Listing 25: Código para la interpolación por el método IDW a todas las bases de datos.

Por otro lado, también se ha querido aplicar el método de IDW^6 con el fin

de aplicar el estudio con ambos métodos y evaluar los resultados con ambos. La aplicación de este segundo método sigue la misma estructura que Lst. 25 modificando la línea de pesos por Lst. 26.

```
pesos <- 1 / distancias^6
```

Listing 26: Modificación para aplicación de IDW^6

Para evaluar el desempeño y la diferencia entre la aplicación de uno u otro método se ha calculado la diferencia relativa entre ambas siguiendo el código en Lst. 27. El resultado es de 0.05278581, es decir, la diferencia relativa entre ambas interpolaciones es de un 5,27 %.

```
480 diff_abs <- abs(as.matrix(predicciones_total_idw) - as.matrix(
        predicciones_total_idw6))
    mean_diff_rel <- mean(diff_abs / as.matrix(predicciones_total_idw6), na
        .rm = TRUE)
482 mean_diff_rel
```

Listing 27: Diferencia relativa entre IDW e IDW^6.

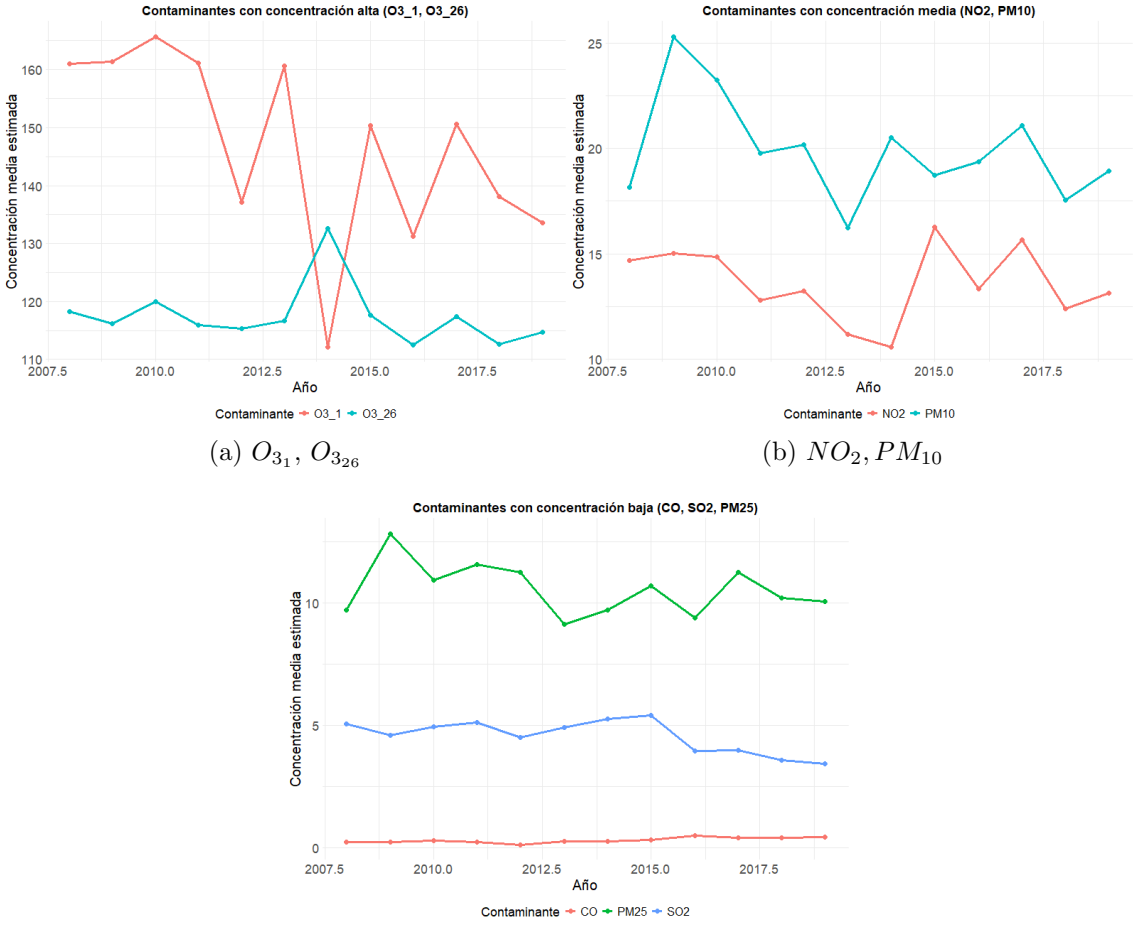
Una vez interpolados los datos se han generado una serie de gráficas que permiten visualizar la evolución temporal de los contaminantes en las zonas donde se ha realizado el estudio. Las gráficas de evolución de las concentraciones de contaminantes en función del año se muestra en Fig.59.

La Fig.59a muestra que los contaminantes O_{3_1}, $O_{3_{26}}$ muestran un comportamiento variable, estas fluctuaciones se pueden deber a cambios meteorológicos como a políticas de control.

La Fig.59b muestra que los contaminantes NO_2, PM_{10} presentan comportamientos diferenciados, mientras que PM_{10} mantiene una evolución constante, NO_2 evidencia una tendencia ligeramente descendente, lo cual podría estar asociado a mejoras en la eficiencia del transporte o medidas sobre emisiones industriales.

La Fig.59c muestra que, los contaminantes CO,SO_2,$PM_{2.5}$ presentan una tendencia ligeramente descendente particularmente clara a partir del año 2015, lo que podría estar relacionado con la transición a combustibles más limpios.

En conjunto, la interpolación permitió obtener los datos de concentración en puntos donde no contábamos con el dato exacto y, además, obtener una visualización robusta de la tendencia de esos contaminantes en los puntos de estudio.



(a) O_{3_1}, $O_{3_{26}}$

(b) NO_2, PM_{10}

(c) CO,SO_2,$PM_{2.5}$

Figura 59: Evolución de los contaminantes en Andalucía a lo largo de los años. Referencias: elaboración propia

6. Modelado predictivo.

En este apartado se presentarán los resultados obtenidos tras la aplicación de diversos algoritmos de Deep Learning para el estudio de la presencia de posibles enfermedades asociadas a la contaminación.

Se evaluaron diferentes arquitecturas, considerando aspectos como el rendimiento, capacidad de generalización y robustez frente a datos ruidosos o incompletos. Para ello, se utilizaron métricas de evaluación diferentes que permitieron comprar el comportamiento de cada modelo.

Asimismo, se describirán los criterios de entrenamiento empleados y los procedimientos de validación cruzada implementados. Esta comparación buscará determinar

que modelos son los más adecuados y obtener conclusiones de los datos.

6.1. Árboles de decisión

Los árboles de decisión presentados en el apartado 3.1.3 y aplicados también a la hora de la interpolación en el apartado 5.6 se utilizarán en este apartado para determinar relaciones entre las variables de la base de datos.

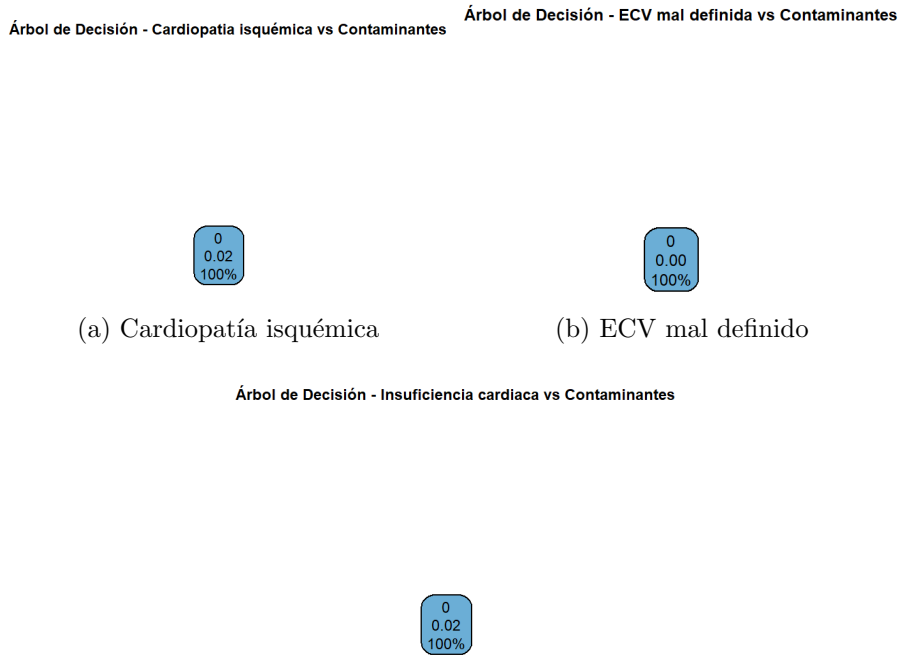
Se partirá de la base de datos pura, es decir, para la aplicación de árboles de decisión se utilizarán todas las variables de enfermedades y contaminantes además de otras variables copredictoras como el sexo (COD_SEXO), la edad (COD_EDAD), el distrito (COD_DISTRITO) y el nivel de ingresos (DESC_TRAMO_FARMACIA). La intención es visualizar sí es posible mejorar el desempeño del programa teniendo en cuenta las concentraciones de contaminantes significando en ese caso que la contaminación podría tener un papel relevante en ciertas enfermedades.

Debido a que la aplicación de árboles de decisión a todas las enfermedades podría suponer mucho tiempo, se aplicará este método únicamente a las enfermedades que, según el estado del arte, pueden estar asociadas con la contaminación. Posteriormente una vez se haya estudiado el desempeño de este método frente al resto se aplicará a todas las enfermedades el método que mejor funcione.

Haciendo uso de las librerías *rpart* y *rpart.plot* se han obtenido los resultados para las enfermedades: diabetes, cardiopatías isquémicas, hipertensión, insuficiencia cardíaca, accidentes cerebro-vasculares (ECV) mal definidos.

Tres de ellos presentaron un único nodo, esto quiere decir que el árbol de decisión indica que no es necesario particionar más. Esto puede deberse por datos muy homogéneos en la base de datos, falta de características con las que partir los datos para mejorar la clasificación o regresión o, que el criterio de parada del árbol no permitió más divisiones. Las enfermedades que han presentado este resultado son: las cardiopatías isquémicas, los accidentes cerebro-vasculares (ECV) mal definidos y la insuficiencia cardíaca. Los resultados gráficos de estos tres casos se muestran en la Fig.60.

Sin embargo, para la diabetes y la hipertensión si que se han encontrado relaciones, estas relaciones aparecen entre diversas enfermedades pero en ningún caso parecen estar afectando los contaminantes. Los resultados se observan en la Fig.61.



(a) Cardiopatía isquémica

(b) ECV mal definido

(c) Insuficiencia cardíaca

Figura 60: Resultado de aplicación de árboles de decisión a algunas enfermedades, sin resultados. Referencias: elaboración propia

El código para la aplicación del método de árboles de decisión se muestra en Lst. 28.

```
 library(rpart)
484 library(rpart.plot)

486 # Asegurar la variable objetivo y variables categ ricas como factor
    dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)
488 dat_salud_idw_mod$COD_DISTRITO <- as.factor(dat_salud_idw_mod$COD_
      DISTRITO)
    dat_salud_idw_mod$COD_SEXO <- as.factor(dat_salud_idw_mod$COD_SEXO)
490 dat_salud_idw_mod$DESC_TRAMO_FARMACIA <- as.factor(dat_salud_idw_mod$
      DESC_TRAMO_FARMACIA)

492 # Dividir los datos en entrenamiento (80%) y prueba (20%)
    set.seed(19354)
494 partici n <- runif(nrow(dat_salud_idw_mod))
    entrenamiento <- dat_salud_idw_mod[partici n < 0.8, ]
496 prueba <- dat_salud_idw_mod[partici n >= 0.8, ]

498 # Crear el modelo del  rbol  de decisi n
```

```
modelo_Diabetes <- rpart(Diabetes ~ ., data = entrenamiento, method = "
    class")

# Graficar
png("Directorio", width = 1000, height = 800)
rpart.plot(modelo_Diabetes, cex = 2)
title(main = "  rbol  de Decisi n - Diabetes vs Contaminantes", cex.
    main = 2)
dev.off()

# Predecir en el conjunto de prueba
predicci n <- predict(modelo_Diabetes, prueba, type = "class")

# Matriz de confusi n
mc <- table(Real = prueba$Diabetes, Predicho = predicci n)
print(mc)

# Calcular la exactitud
exactitud <- sum(diag(mc)) / sum(mc)
cat("Exactitud en conjunto prueba:", round(exactitud * 100, 2), "%\n")
```

Listing 28: Código para la aplicación de árboles de decisión

Los resultados obtenidos en la matriz de confusión se muestra en la Tab.15

Real \Predicho	**0**	**1**
0	149363	718
1	12926	1319
Exactitud	91.64 %	
Precisión (Clase 1)	64.76 %	
***Recall* (Clase 1)**	9.26 %	

Tabla 15: Matriz de confusión – Diabetes (árboles de decisión).

La exactitud es de 91.7 % esto indica que los resultados presentan un buen desempeño. Sin embargo, el modelo parece clasificar bien para los casos negativos (no diabetes), por otro lado, parece haber una cantidad considerable de falsos negativos, es decir, pacientes con diabetes que el modelo predice erróneamente.

6.1.1. Balanceo de datos

Habiendo visto los resultados de aplicación de los árboles de decisión y el desempeño del método parece haber un desbalanceo en el número de datos de diabetes positiva y negativa, esto puede haber afectado a los resultados y, por tanto, se van a probar algunos métodos diferentes para balancear los datos y analizar cuáles son los resultados.

Primero se aplicará un balanceo por submuestreo, esto quiere decir, se eliminarán los datos de la clase mayoritaria e igualar así la cantidad con la clase minoritaria. El riesgo es, que al perder datos verdaderos, el modelo podría generalizar peor. Para aplicar el submuestreo se ha añadido al código anterior las líneas de código mostradas en Lst. 29.

```
# Submuestreo de la clase mayoritaria en el conjunto de entrenamiento
clase0 <- entrenamiento_original %>% filter(Diabetes == "0")
clase1 <- entrenamiento_original %>% filter(Diabetes == "1")
n_min <- min(nrow(clase0), nrow(clase1))
set.seed(123)
clase0_sub <- clase0 %>% sample_n(n_min)
clase1_sub <- clase1 %>% sample_n(n_min)
entrenamiento_balanceado <- bind_rows(clase0_sub, clase1_sub)

# Revision del balance
table(entrenamiento_balanceado$Diabetes)
```

Listing 29: Submuestreo

El resultado del modelo árbol en esta ocasión se muestra en Fig.62. Y la matriz de confusión en la Tab.16. La exactitud será en esta ocasión del 73.95 %.

En el modelo balanceado (Fig.62), el árbol de decisión revela que, además de la edad y la presencia de hipertensión, la variable DESC_TRAMO_FARMACIA (indicativa del tipo de aseguramiento farmacéutico) tiene un peso relevante en la predicción de la diabetes. Esta variable, que agrupa a los individuos según su condición de pensionista o activo y su nivel de renta, se mostró especialmente discriminante en el subconjunto sin hipertensión y edad inferior a 53 años. Esta observación resalta la importancia del perfil socioeconómico en la prevalencia de la diabetes, ya que ciertos tramos como los pensionistas con menor renta (TSI 001) presentaron una mayor proporción de casos positivos.

Real \Predicho	**0**	**1**
0	113950	36227
1	2443	12034

Tabla 16: Matriz de confusión – Datos balanceados

Los resultados obtenidos del balanceo se evaluarán también mediante los siguientes métodos:

- Exactitud.
- Sensibilidad (*Recall o TPR*) para la clase de diabetes positiva: Mide la proporción de los casos positivos fueron detectados. Se calcula siguiendo la Eq. 71

$$Recall = \frac{TP}{TP + FN} \tag{71}$$

- Precisión para la clase de diabetes positiva: Mide qué proporción de las predicciones positivas fueron correctas.

$$Precision = \frac{TP}{TP + FP} \tag{72}$$

Donde:

- TP (*True Positive*: son los casos realmente positivos y predichos como positivos.
- FP (*False Positive*: son los casos realmente negativos y predichos como positivos.
- TN (*True Negative*: son los casos realmente negativos y predichos como negativos.
- FN (*False Negative*: son los casos realmente positivos y predichos como negativos.

Los resultados obtenidos en estos cálculos se muestran en la Tab.**??**.

Como se puede observar en la tabla de resultados la exactitud ha empeorado pero hay que tener en cuenta que la exactitud en el caso de desbalanceado dependía

Métrica	Balanceado	Desbalanceado
Exactitud (Accuracy)	73.39 %	91.87 %
Sensibilidad (Recall)	83.11 %	9.22 %
Precisión (Precisin)	24.94 %	64.75 %

Tabla 17: Métricas comparadas – Modelo balanceado vs. desbalanceado.

también de que el número de casos negativos de diabetes era muy alto. Por otro lado, la sensibilidad parece ser mejor en los casos positivos, es decir, encuentra la mayoría de personas con diabetes. Por último, la precisión parece ser mayor en el caso del modelo desbalanceado a costa de casi no detectar muchos casos reales.

En conclusión, el modelo balanceado mejora mucho la detección de la clase minoritaria (personas con diabetes), aumentando el *Recall*. Esto es clave en problemas médicos donde detectar casos positivos es vital aunque haya más falsos positivos.

Se ha optado por aplicar el balanceo de datos en aquellas enfermedades cuyas predicciones eran nulas o no se obtenían resultados válidos. Los resultados se muestran en la Fig.63.

Los resultados de las matrices de confusión para los tres casos se muestra en la Tab.18.

Condición	Matriz de Confusión		Exactitud (%)
	0	**1**	
ECV Mal definida	122090	42164	74.36
	55	345	
	0	**1**	
Cardiopatía Isquémica	124026	36856	77.38
	390	3383	
	0	**1**	
Insuficiencia Cardíaca	131059	29550	81.73
	534	3511	

Tabla 18: Matrices de confusión – Enfermedades cardiovasculares.

6.2. Bosques aleatorios

En esta sección se aplicarán bosques aleatorios a las enfermedades más relevantes. El fin de este apartado será determinar si mejora la exactitud del método aplicando o sin aplicar los contaminantes. En caso de que se viera mejora de la exactitud con

los datos de contaminantes se consideraría que éstos son relevantes para el estudio de ciertas enfermedades.

6.2.1. Diabetes

El primer caso de aplicación sería para el caso de la diabetes. Con el fin de disminuir el tiempo se entrenará el bosque con 100 árboles aleatorios ya que, el valor definido por defecto (500 árboles) podría llegar a suponer hasta 3 horas de procesamiento. Con 100 árboles de decisión el tiempo de procesado ha sido de entre 10-30 minutos.

El código para la aplicación del modelo se muestra en Lst. 30. Para la aplicación de este se han hecho uso de las librerías *Random Forest* y *data.tree*, éstas permiten aplicar clasificación y regresión a nuestros datos y representarlos siendo muy robustos frente a *overfitting*.

```
528 library(randomForest)
    library(rpart.plot)
530 library(data.tree)

532 set.seed(123)

534 # Asegurarse que la variable resultado es un factor
    dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)
536
    # Dividir datos en entrenamiento (85%) y prueba (15%)
538 particion <- runif(nrow(dat_salud_idw_mod))
    entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
540 prueba <- dat_salud_idw_mod[particion >= 0.85, ]

542 # Entrenar modelo random forest
    modelo_rf <- randomForest(Diabetes ~ ., data = entrenamiento, ntree =
        100, importance = TRUE)
544 print(modelo_rf) # Resumen del modelo

546 print(importance(modelo_rf)) # Mostrar variables mas importantes
    varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
548
    predi_rf <- predict(modelo_rf, prueba) # Predecir en conjunto de prueba
550
    mc_rf <- table(Predicci n = predi_rf, Real = prueba$Diabetes) # Matriz
```

```
    de confusi n
print(mc_rf)

# Calcular exactitud (accuracy)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")

png("Directorio",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
dev.off()
```

Listing 30: Código para la aplicación de bosques aleatorios para la enfermedad: diabetes.

La matriz de confusión obtenida en esta primera prueba con la diabetes se muestra en Tab.19.

Real \Predicho	**0**	**1**
0	111111	8790
1	1812	2032
Exactitud	91.43 %	
Precisión (Clase 1)	18.74 %	
***Recall*(Clase 1)**	52.88 %	

Tabla 19: Matriz de confusión – Diabetes con contaminantes (bosques aleatorios).

En general, se puede ver una exactitud bastante buena pero, como ya se vio en árboles de decisión, muchos errores en la predicción de casos positivos de diabetes donde el porcentaje de acierto es del 18.8 %.

El análisis de importancia de variables Fig.64 ha permitido identificar los factores más relevantes asociados con el diagnóstico de la diabetes. En esta gráfica se presentan dos métricas: la disminución de la precisión (*MeanDrecreaseAccuracy*) y la disminución del índice de Gini (*MeanDrecreaseGini*).

En la gráfica se puede observar que la edad parece ser una variable importante en ambas métricas al igual que la hipertensión y la obesidad. En cuanto al estudio, las variables ambientales ($SO_2, O_3, NO_2, PM_{10}, PM_{2.5}$) parecen mostrar cierta relevancia lo cual sugiere una posible relación entre la exposición a contaminantes y el riesgo de desarrollar diabetes.

Aunque con la gráfica de importancia parece verse cierta relación también he

querido probar eliminando las variables de contaminantes para ver así la exactitud del modelo. Para ello se ha añadido solo la siguiente línea de código (Lst. 31) y se ha modificado el resto del código acorde a esta nueva variable.

```
562 dat_sin_contaminantes <- dat_salud_idw_mod[, !(names(dat_salud_idw_mod)
      %in% c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2"))]
```

Listing 31: Eliminación de las variables de contaminantes en la base de datos

El resultado de eliminar los contaminantes de la base de datos en esta ocasión han dado los resultados mostrados en la Tab.20.

Real \Predicho	**0**	**1**
0	111681	8977
1	1242	1845
Exactitud	91.74 %	

Tabla 20: Matriz de confusión – Diabetes sin contaminantes (bosques aleatorios).

A primera vista parece incluso que el añadir las variables de contaminantes a la base de datos la exactitud empeora además, analizando más en profundidad los datos se puede observar que:

- Con contaminantes los falsos negativos parecen ser más (1812) que en el caso sin contaminantes (1242).
- Con contaminantes los falsos positivos (8790) parecen ser menos que en el caso de sin contaminantes (8977).
- Con contaminantes el *Recall* para los casos positivos es de 52.86 % mientras que sin ellos es de 59.76 %.

Con todo esto se puede observar que sin contaminantes parece que el método actúa de manera más conservadora, es decir, prioriza dar falsos positivos que falsos negativos, lo cual, a ojos de la medicina puede ser positivo. Además, la exactitud parece mejorar.

En conclusión, el hecho de que una variable tenga alta importancia no garantiza que su presencia mejore la exactitud del modelo en test. Puede ser ruido, puede estar correlacionada con otras, o puede inducir overfitting. Lo importante es evaluar su efecto real sobre la generalización, no solo su uso interno.

6.2.2. Dislipemia

La dislipemia es un trastorno del metabolismo de los lípidos en la sangre. Se trata de un factor de riesgo para enfermedades cardiovasculares.

Ciertos estudios recientes han demostrado que la exposición crónica a contaminantes atmosféricos puede aumentar el riesgo de dislipemia ya que, en general, la contaminación puede estar asociada con un aumento de los niveles de colesterol total.

La ejecución del código para el caso de la Dislipemia ha arrojado los resultados que se muestran en Tab.21 y la Fig.65.

Real \Predicho	**0**	**1**
0	88221	14566
1	7902	13056
Exactitud	81.84 %	

Tabla 21: Matriz de confusión – Dislipemia con contaminantes (bosques aleatorios).

Se ha probado además modificando la base de datos para eliminar la presencia de otras enfermedades y dejar únicamente los contaminantes para ver qué información arrojaba. Los resultados se muestran en la Tab.23 y la Fig.66

Real \Predicho	**0**	**1**
0	96103	27601
1	20	21
Exactitud	77.68 %	

Tabla 22: Matriz de confusión – Dislipemia (solo contaminantes, bosques aleatorios).

A pesar de que, aparentemente la exactitud es buena, parece que no realiza correctamente predicciones en el caso de positivos en dislipemia. Lo mismo ocurre con el resto de enfermedades donde, a pesar de que la exactitud es cercana al 100 % esta se basa en la no-predicción, es decir, considera todos los casos negativos ya que la mayor parte de los datos pertenecen a ese grupo.

Por otro lado, también se ha aplicado el balanceo de clases a esta misma casuística. Los resultados se muestran en la Tab.23 y la Fig.67.

Como se puede observar, con los datos balanceados sí que se realizan predicciones para positivos en dislipemia pero el valor de la exactitud es cercano a tirar una

Real \Predicho	**0**	**1**
0	57626	13946
1	38497	13676
Exactitud	57.62 %	

Tabla 23: Matriz de confusión – Dislipemia (contaminantes + datos balanceados).

moneda al aire (50 %) esto indica que el modelo no predice bien y, por tanto, no se puede asumir a ciencia cierta que, tal y como se muestra en la Fig.67 el O_{3_1} o el $PM_{2.5}$ tengan una afectación real en el caso de la dislipemia.

6.2.3. Trastorno de ánimo

La exposición a contaminantes del aire se ha asociado con un aumento de síntomas de ansiedad, depresión y estrés [77]. La ejecución del código para este caso se muestra en la Tab.24 y Fig.68.

Real \Predicho	**0**	**1**
0	117049	6053
1	397	246
Exactitud	94.79 %	

Tabla 24: Matriz de confusión – Trastorno de ánimo (con contaminantes).

Se han realizado pruebas, al igual que en el caso de la dislipemia, con balanceo de datos y analizando únicamente la presencia de contaminantes con el fin de poder estudiar relaciones entre éstos y la enfermedad. El resultado de la exactitud ha vuelto a ser en torno al 50 % (52.68 %), esto quiere decir que, con únicamente la presencia de contaminantes no se pueden asumir relaciones entre éstos y el trastorno de ánimo.

Con el caso de la no aplicación del balanceo de datos y la eliminación de variables de tipo enfermedad la exactitud era de un 94.91 %, aunque esta, una vez más, no es significativa ya que no está produciendo ninguna predicción para los casos de trastorno de ánimo positivo.

6.2.4. Trastorno de ansiedad

Al igual que en el caso anterior se pueden encontrar estudios que reflejan relaciones entre los trastornos de ansiedad y el aumento de la contaminación. La ejecución del código para este caso se muestra en la Tab.25 y la Fig.69.

Real \Predicho	0	1
0	109116	11578
1	1400	1651
Exactitud	89.51 %	

Tabla 25: Matriz de confusión – Trastorno de ansiedad (con contaminantes).

6.2.5. Arteriopatía periférica de extremidades inferiores (APEI)

La arteriopatía periférica de extremidades inferiores es una enfermedad caracterizada por el estrechamiento u obstrucción de las arterias que suministran sangre a las piernas. Como ya se ha estudiado la exposición a contaminantes atmosféricos está asociada con un aumento de las enfermedades cardiovasculares.

Los resultados de aplicación del código para este caso se muestran en la Tab.26 y la Fig.70.

Real \Predicho	0	1
0	121062	2591
1	61	31
Exactitud	97.86 %	

Tabla 26: Matriz de confusión – Arteriopatía de extremidades (con contaminantes).

En el caso del trastorno de ansiedad al igual que en caso del trastorno de ánimo, el resultado de eliminación del resto de enfermedades en la base de datos da una exactitud de 89.31 % a pesar de que esta no refleja la realidad ya que no realiza predicciones sobre los casos de ansiedad positivos. Además, al aplicar el balanceo de clases, el resultado de la exactitud es del 50.95 %, es decir, irrelevante.

6.2.6. Cardiopatías isquémicas

Las cardiopatías isquémicas al igual que la arteriopatía de extremidades guarda relación con las enfermedades cardiovasculares. Se trata de una condición caracterizada por el estrechamiento u obstrucción de las arterias coronarias.

Los resultados del estudio en este caso se muestra en la Tab.27 y la Fig.71.

Real \Predicho	**0**	**1**
0	120710	2543
1	264	228
Exactitud	97.73 %	

Tabla 27: Matriz de confusión – Cardiopatías isquémicas (con contaminantes).

6.2.7. Hipertensión

La hipertensión arterial es una condición caracterizada por la elevación sostenida de la presión arterial. La exposición a contaminantes se ha asociado con un aumento en la incidencia y gravedad de la hipertensión arterial.

El estudio sobre esta enfermedad arroja los siguientes resultados: Tab.30 y Fig.

Real \Predicho	**0**	**1**
0	91043	10987
1	6140	15575
Exactitud	86.16 %	

Tabla 28: Matriz de confusión – Hipertensión (con contaminantes).

En el caso de la hipertensión se intentó de nuevo realizar el balanceo de datos y eliminar todas las variables salvo los contaminantes ya que se podría esperar mejores resultados por la presencia de un mayor número de pacientes con hipertensión. Los resultados se muestran en la Tab.29. Como se puede observar una exactitud cercana al 50 %, por tanto, no se pueden obtener resultados concluyentes de este modelo.

Real \Predicho	**0**	**1**
0	49935	11557
1	47248	15005
Exactitud	52.48 %	

Tabla 29: Matriz de confusión – Hipertensión (contaminantes + datos balanceados).

6.2.8. Insuficiencia cardíaca

Al igual que en casos anteriores la insuficiencia cardíaca se ve asociada a la exposición a diferentes contaminantes. Los resultados del estudio sobre esta variable se muestran en Tab.30 y Fig.73.

Real \Predicho	**0**	**1**
0	120440	2760
1	299	246
Exactitud	97.53 %	

Tabla 30: Matriz de confusión – Insuficiencia cardíaca (con contaminantes).

6.2.9. Balanceo de datos

La aplicación del balanceo de datos para la diabetes con contaminantes haciendo uso del mismo método aplicado en Lst.29 ha arrojado los siguientes datos aplicando el balanceo con la base de datos que contiene contaminantes (Tab.31) y sin contaminantes (Tab.32).

Real \Predicho	**0**	**1**
0	88503	1873
1	24420	8949
Exactitud	78.75 %	

Tabla 31: Matriz de confusión – Diabetes (contaminantes + datos balanceados).

Real \Predicho	**0**	**1**
0	88804	1708
1	24119	9114
Exactitud	79.13 %	

Tabla 32: Matriz de confusión – Diabetes (sin contaminantes, datos balanceados).

Una vez más, en el caso de añadir los contaminantes a la base de datos parece que, en la valoración de la importancia de cada variable (Fig.74) los contaminantes parecen tener alguna importancia en la evolución de la exactitud, más en concreto PM_{10}. Sin embargo, en el análisis de la matriz de confusión parecen funcionar mejor la base de datos en la que no se tienen en cuenta los contaminantes.

Además, se puede volver a apreciar que, pese a que el balanceo empeore la exactitud el *recall* de la variable de diabetes positiva parece mejorar desde un 59.76 % hasta un 84.23 %. Aunque, al igual que en el caso de árboles de decisión, el número de falsos positivos también es mayor.

6.2.10. Aplicación de la base de datos con IDW^6

En todos los casos hasta ahora se ha aplicado el método de IDW ya que éste es el método más extendido y el que mayor seguridad generaba pese a haber encontrado mejor funcionamiento con el uso de la interpolación IDW^6.

Con los contaminantes interpolados por el método IDW^6 los resultados se muestran en la Tab.33 y en la Fig.75. Los modelos aplicados a la base de datos con los contaminantes interpolados mediante el método de IDW^6 aparentan reflejar un mejor funcionamiento del modelo e, incluso, en la Fig.75 se puede apreciar una mucha mayor importancia de los contaminantes en las predicciones pasando de estar en la cuarta posición en la Fig.64 donde los datos se interpolaron con el método IDW a aparecer en tercera posición con la interpolación por el método de IDW^6.

Real \Predicho	**0**	**1**
0	111106	8731
1	1817	2091
Exactitud	91.48 %	
Precisión (Clase 1)	19.32 %	
***Recall*(Clase 1)**	53.49 %	

Tabla 33: Matriz de confusión – Diabetes (contaminantes + IDW^6).

Sin embargo, el método con únicamente contaminantes parece volver a no predecir bien ya que no realiza predicciones positivas en ningún caso.

Por otro lado, si se comparan los resultados en este caso con los resultados sin tener en cuenta los contaminantes la precisión parece mejorar (lo que quiere decir que se predicen mejor los casos positivos) pero empeora ligeramente el *recall*. La exactitud en términos generales también será mejor en el caso de no tener en cuenta los contaminantes.

6.3. Redes neuronales

Con el fin de poder comparar entre diversos métodos se ha optado por analizar también el desempeño del método de Redes Neuronales. Para la aplicación de este método se ha aplicado el código mostrado en Lst. 32. Este código hace uso de la librería específica para la aplicación de redes neuronales *nnet*.

```
library(caret)
```

```
564 library(ggplot2)
    library(lattice)
566
    set.seed(123)
568
    # Asegurarse que la variable respuesta es factor
570 dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)

572 # Escalar las variables num ricas
    preproc <- preProcess(dat_salud_idw_mod[, -which(names(dat_salud_idw_
        mod) == "Diabetes")], method = c("center", "scale"))
574 dat_scaled <- predict(preproc, dat_salud_idw_mod)
    dat_scaled$Diabetes <- dat_salud_idw_mod$Diabetes  # A adir la
        variable respuesta
576
    # Dividir datos en entrenamiento (85%) y prueba (15%)
578 particion <- runif(nrow(dat_scaled))
    entrenamiento <- dat_scaled[particion < 0.85, ]
580 prueba <- dat_scaled[particion >= 0.85, ]

582 # Entrenar red neuronal
    modelo_nn <- nnet(Diabetes ~ ., data = entrenamiento, size = 5, maxit =
         200, decay = 0.01, trace = FALSE)
584
    # Predecir en conjunto de prueba
586 predi_nn_prob <- predict(modelo_nn, prueba, type = "raw")
    predi_nn <- ifelse(predi_nn_prob > 0.5, "1", "0")
588 predi_nn <- factor(predi_nn, levels = levels(prueba$Diabetes))

590 # Matriz de confusi n
    mc_nn <- confusionMatrix(predi_nn, prueba$Diabetes)
592 print(mc_nn$table)

594 # Exactitud
    cat("Exactitud en conjunto prueba:", round(mc_nn$overall["Accuracy"] *
        100, 2), "%\n")
```

Listing 32: Aplicación del método de redes neuronales para la diabetes como variable objetivo.

Se ha hecho uso de la herramienta *Chat GPT* de *OpenAI* para aplicar este código dando como referencia el de árboles aleatorios. Las redes han sido entrenadas con cinco neuronas en la capa oculta, un máximo de 200 iteraciones, un *decay* (evita el

sobreajuste) de 0.01.

En la Tab.34 se muestran los resultados de este estudio.

Real \Predicho	**0**	**1**
0	111597	8872
1	1326	1950
Exactitud	91.76 %	
Precisión (Clase 1)	18.02 %	
***Recall* (Clase 1)**	59.55 %	

Tabla 34: Matriz de confusión – Diabetes (redes neuronales, con contaminantes).

Dejando únicamente los parámetros de contaminantes los resultados se muestran en la Tab.35.

Real \Predicho	**0**	**1**
0	112923	10822
1	0	0
Exactitud	91.25 %	
Precisión (Clase 1)	0.00 %	
***Recall* (Clase 1)**	–	

Tabla 35: Matriz de confusión – Diabetes (solo contaminantes, redes neuronales).

Y al eliminar los contaminantes y dejar solo las enfermedades los resultados se muestran en Tab.36.

Real \Predicho	**0**	**1**
0	111019	8528
1	1904	2294
Exactitud	91.57 %	
Precisión (Clase 1)	21.19 %	
***Recall*(Clase 1)**	54.63 %	

Tabla 36: Matriz de confusión – Diabetes (sin contaminantes, redes neuronales).

El análisis comparativo de los modelos de redes neuronales para la predicción de diabetes, con y sin inclusión de variables relacionadas con contaminantes, muestra diferencias relevantes en sus métricas de desempeño.

El modelo que incorpora los contaminantes presenta una ligera mejora en la exactitud global (91.76 % frente a 91.57 %) y un aumento significativo en el *Recall* para la clase positiva (59.55 % frente a 54.63 %). Esto indica que este modelo es

más eficaz para identificar correctamente a los pacientes con diabetes, reduciendo la cantidad de casos positivos no detectados. Sin embargo, esta mayor sensibilidad se acompaña de una menor precisión (18.02 % frente a 21.19 %), lo que implica un incremento en los falsos positivos.

Por otro lado, el modelo sin variables de contaminantes ofrece una mayor precisión, es decir, sus predicciones positivas son más confiables, pero a costa de perder capacidad para detectar todos los casos reales de diabetes.

Por último, se ha realizado la prueba con la base de datos de contaminantes interpolados por el método de IDW[6], en este caso los resultados han sido exactamente los mismos que los mostrados en la Tab.36 parece que, en este caso, el método no es tan sensible a pequeñas variaciones en la interpolación.

6.4. Conclusiones

Modelo	**TN**[4]	**FP**[5]	**FN**[6]	**TP**[7]	**Exactitud**
AD[8]	149363	718	12926	1319	91.70 %
AD balanceado	113950	36227	2443	12034	73.39 %
BA[9] con contaminantes	111111	8790	1812	2032	91.43 %
BA sin contaminantes	111681	8977	1242	1845	91.74 %
BA sin contaminantes, balanceado	88804	1708	24119	9114	79.13 %
BA con contaminantes, balanceado	88503	1873	24420	8949	78.75 %
BA con contaminantes (IDW[6])	111106	8731	1817	2091	91.48 %
NNET[10] con contaminantes	111597	8872	1326	1950	**91.76 %**
NNET con únicamente cont.	112923	10822	0	0	91.25 %
NNET sin contaminantes	111019	8528	1904	2294	91.57 %

Tabla 37: Comparativa de modelos – Árboles, bosques y redes neuronales.

Analizando los resultados de aplicación de diferentes algoritmos de *Machine Learning* mostrados en la Tab.37 y Tab.38. Se han obtenido diversas conclusiones.

[4]TN (true negative): clase 0 predicha como 0
[5]FP (false positive): clase 0 predicha como 1
[6]FN (false negative): clase 1 predicha como 0
[7]TP (true positive: clase 1 predicha como 1
[8]AD: árbol de decisión
[9]BA: bosques aleatorios
[10]NNET: redes neuronales
[11]Precisión: TP / (TP + FP)
[12]*Recall*: TP / (TP + FN)

Modelo	**Precisión**[11]	***Recall***[12]
AD	64.74 %	9.26 %
AD balanceado	24.94 %	**83.12 %**
BA sin contaminantes	17.05 %	59.77 %
BA con contaminantes	18.78 %	52.85 %
BA sin contaminantes, balanceado	**84.23 %**	27.42 %
BA con contaminantes, balanceado	82.69 %	26.81 %
BA con contaminantes (IDW^6)	19.32 %	53.49 %
NNET con contaminantes	18.02 %	59.55 %
NNET con únicamente cont.	0 %	—
NNET sin contaminantes	21.19 %	54.63 %

Tabla 38: Precisión y sensibilidad – Diabetes (árboles de decisión y bosques aleatorios).

En referente a los árboles de decisión es notable el mejor funcionamiento del método para la detección de casos positivos en el caso del balanceo de datos. Sin embargo, disminuye drásticamente el *Recall*.

En cuanto a los bosques aleatorios vuelve a suceder lo mismo con el caso de modelos balanceados. Y, además, parece que incluir contaminantes en la base de datos reduce ligeramente el *Recall* aunque aumenta la precisión. El modelo con la interpolación IDW^6 parece lograr un equilibrio ligeramente mejor en términos de precisión y *Recall*.

Las redes neuronales con contaminantes parecen lograr el mejor balance entre precisión y *Recall* entre todos los modelos no balanceados. Al eliminar los contaminantes de la base de datos parece que se mejora la precisión pero el *Recall* es más bajo.

En general, se observan diferencias notables entre cada uno de los métodos. Los árboles de decisión presentan un rendimiento desigual, el *Recall* parece ser extremadamente bajo, lo que los hace poco útiles y en general algo menos inestables. Por su parte, los bosques aleatorios requieren de un mayor tiempo de procesamiento, logran mayor exactitud que el caso de árboles de decisión pero se sigue pudiendo observar un desbalanceo aparente entre precisión y *Recall*. Por último, las redes neuronales ofrecen un mejor equilibrio en general, especialmente contando con los contaminantes en la base de datos, además, el tiempo de ejecución de este método es algo menor que en caso de los bosques aleatorios.

7. Conclusiones y lineas futuras

El presente proyecto tenía como objetivo general el desarrollar, mediante diferentes técnicas de *Machine Learning* haciendo énfasis en los árboles de decisión, un sistema capaz de analizar y predecir el impacto de la contaminación ambiental en la salud de la población. A partir de este planteamiento se han abordado diversas etapas.

7.1. Consecución de los objetivos

En primer lugar se ha realizado una revisión previa al proyecto que ha permitido comprender mejor las conclusiones a las que estudios científicos previos habían llegado sobre posibles afectaciones de la contaminación ambiental en la salud. Además, ha permitido obtener información valiosa sobre diferentes métodos de interpolación y aprendizaje automático las cuales han facilitado el proceso de aprendizaje y elaboración del presente proyecto.

A continuación, se procedió a la importación y tratamiento previo, asegurando su correcta integración y homogeneización de las variables incluidas en el estudio. Esa etapa incluyó la transformación de formatos, detección de inconsistencias o valores anómalos y limpieza de los datos con el fin de garantizar la idoneidad para su posterior uso en los distintos métodos de aplicación.

Posteriormente, se aplicaron diversas técnicas de interpolación espacial con el objetivo de estimar los valores de contaminación ambiental en aquellas ubicaciones donde no se disponían de datos directos, pero sí que se tenían parámetros de salud. También se llevó a cabo una evaluación comparativa del rendimiento de cada técnica de interpolación en la base de datos que se tenía.

Por último, se llevó a cabo la implementación y comparación de distintos algoritmos de *Machine Learning* con el objetivo de evaluar su capacidad para modelar la relación entre la contaminación ambiental y distintos parámetros de salud. La finalidad de este apartado sería encontrar relaciones entre estas las variables ambientales y los indicadores sanitarios, y estudiar el desempeño de diversos enfoques metodológicos sobre la base de datos previamente preparada.

7.2. Validación de las hipótesis

Tras la implementación de los modelos y análisis de los resultados, se procede a evaluar el grado de cumplimiento de las hipótesis planteadas al inicio del proyecto.

Respecto a la **Hipótesis 1**, que proponía una relación positiva entre las concentraciones de agentes contaminantes y la incendiaria de enfermedades respiratorias y cardiovasculares, los resultados obtenidos parecen no mostrar una correlación clara ni consistente. De hecho, en algunos casos, la inclusión de las concentraciones de contaminantes parecen empeorar el desempeño de los modelos de aprendizaje automático.

En cuanto a la **Hipótesis 2**, referida al rendimiento de los algoritmos de *Machine Learning*, los resultados muestran que el modelo de *Random Forest* ofrece un desempeño notable, superando a técnicas como árboles de decisión simples en términos de precisión y estabilidad. No obstante, las redes neuronales presentaron un rendimiento ligeramente superior en algunas métricas. Esto sugiere que, si bien *Random Forest* es una buena opción, las técnicas de redes neuronales pueden ofrecer una mejor capacidad de predicción.

Por último, respecto a la **Hipótesis 3**, los experimentos realizados conforman que el método de interpolación espacial *Inverse Distance Weighting* IDW es eficaz para estimar los valores de contaminación en zonas sin datos directos. Entre las variantes de éste probadas, aparentemente el método IDW^6 demostró un rendimiento superior, proporcionando estimaciones coherentes con tiempos reducidos.

7.3. Futuras líneas de investigación

A partir de los resultados obtenidos en este proyecto, se proponen varias líneas de investigación que podrían ser exploradas en el futuro con el objetivo de poder profundizar y mejorar la capacidad predictiva de los modelos:

- **Ampliación geográfica de la base de datos:** incluir datos de otras regiones permitiría aumentar la variabilidad en las concentraciones de contaminantes y, de esta manera, ayudar a detectar patrones más claros en su relación con las enfermedades estudiadas.
- **Estudio específico por enfermedad:** abordar cada una de las enfermeda-

des de manera individual, así, se podrá mejorar el ajuste de cada uno de los modelos a cada una de las enfermedades permitiendo un análisis más exacto y preciso.

- **Análisis de la temporalidad:** no se han tenido en cuenta las fechas a la hora de realizar el análisis pese a estar presentes en la base de datos. Se propone estudiar la evolución de estos contaminantes y establecer patrones temporales.
- **Incorporación de otras enfermedades:** aparentemente la contaminación afecta principalmente a enfermedades respiratorias no presentes en la base de datos. Se propone añadir este tipo de enfermedades y parámetros a la base de datos.
- **Aplicación del estudio a otras enfermedades:** debido a la naturaleza y alcance de los datos disponibles, no ha sido posible estudiar todas las enfermedades de interés. Por ello, se propone extender este mismo análisis a otras patologías relevantes en futuras investigaciones.

Estas propuestas podrían contribuir a una comprensión más profunda de los factores que afectan a la salud poblacional y a una mejora en la toma de decisiones en políticas de salud pública.

7.4. Reflexiones finales

Se ha observado que, para los datos distribuidos de forma regular en una malla, los métodos de interpolación que han mostrado un peor desempeño son los árboles de decisión, tanto por su precisión como por su sensibilidad al ruido. Por otro lado, el método IDW (*Inverse Distance Weighting*) ha demostrado ser el más eficiente, mostrando un buen funcionamiento con tiempos reducidos.

En general, los árboles de decisión no han alcanzado una precisión comparable a la de modelos como bosques aleatorio o redes neuronales. El desbalanceo de clases (positivos y negativos) ha afectado negativamente la capacidad de los modelos para predecir casos positivos en la mayoría de enfermedades analizadas.

Aunque los gráficos de importancia de variables sugiere una relación entre contaminantes atmosféricos y las enfermedades estudiadas, la eliminación de estas variables parece mejorar la precisión de los modelos. Esto sugiere que la presencia de éstos

podría estar introduciendo ruido, en lugar de aportar un valor predictivo. Además, al limitar los datos geográficos únicamente a Andalucía, la baja variabilidad en la concentración de contaminantes podría estar limitando su utilidad como variable explicativa.

Se han analizado diferentes enfermedades: diabetes, dislipemia, trastorno de ánimo, trastorno de ansiedad, arteriopatía periférica de extremidades inferiores, cardiopatías isquémicas, hipertensión e insuficiencia cardíaca. En ninguno de los casos se ha observado una mejora del modelo al incluir contaminantes en la base de datos. Por otro lado, realizar el balance de los datos mejora la predicción para detectar casos positivos de diabetes, aunque a costa de una reducción en la exactitud global del modelo.

Asimismo, se ha comprobado que eliminar las variables relacionadas con enfermedades y dejar únicamente los contaminantes como predictores resultad ineficiente, ya que el modelo pierde completamente su capacidad predictiva.

En conjunto, el estudio evidencia que, la elección del método de interpolación y el tratamiento de los datos desbalanceados son aspectos críticos en la calidad de los modelos. Además, aunque inicialmente, los contaminantes atmosféricos parecían relevantes, su impacto real sobre la precisión de los modelos ha sido limitado, probablemente por la homogeneidad geográfica de los datos disponibles. Estos hallazgos ofrecen una base sólida para futuras investigaciones más amplias, que puedan incluir mayor diversidad geográfica y variables adicionales para contrastar los resultados obtenidos.

7.4.1. Conclusiones personales

Uno de los principales retos enfrentados durante el desarrollo del trabajo ha sido el procesamiento y manejo del código debido al tamaño de las bases de datos utilizadas. Este factor provocaba un aumento considerable en los tiempos de carga y ejecución, dificultando así la experimentación y la iteración con diferentes modelos y parámetros.

Por otro lado, la calidad y homogeneidad limitada de los datos, junto a la restricción geográfica, condicionaron en gran medida los resultados obtenidos, limitando la capacidad de generalización y robustez de los modelos.

Este proyecto ha sido una experiencia de gran valor para mi formación técnica y

personal. Ha mejorado mis habilidades en el manejo de grandes volúmenes de datos y en la aplicación de técnicas avanzadas de interpolación y aprendizaje automático. Al mismo tiempo, el proceso me ha enseñado a gestionar de manera eficiente los recursos computacionales y a ser paciente y metódico en la interpretación de resultados complejos. A su vez, he podido mejorar y reforzar mi capacidad crítica para evaluar los datos y resultados.

Aunque los resultados obtenidos parecen evidenciar ciertas limitaciones, este estudio puede contribuir a la aportación de información relevante para comprender la relación entre la contaminación atmosférica y la salud, especialmente en contextos con características geográficas semejantes.

Considero que esta investigación contribuirá a visibilizar la importancia de contar con bases de datos amplias y diversas, además de servir como guía para futuros investigadores que deseen retomar y ampliar este estudio.

Por último, la publicación de resultados sólo cuando confirman una hipótesis lleva a una aberración estadística: Puesto que los fenómenos estudiados son usualmente de naturaleza estocástica. Si varios investigadores experimentan sobre algo, un porcentaje bajo de ellos obtendrá un resultado diferente a la moda. Por la naturaleza de las revistas, se tienden a publicar los resultados sorprendentes o que confirman las creencias. Puede darse entonces el caso que se publiquen las condiciones obtenidas en los experimentos que se salen de la "normalidad", creándose creencias falsas. Es por ello que es tan importante publicar tanto los resultados "positivosçomo los "negativos". En este caso se ha observado que las variables de los contaminantes no son muy importantes a la hora de predecir las enfermedades estudiadas. Este es el resultado y así se refleja en esta memoria.

Bibliografía

[1] Manuel Romero Placeres, Francisca Diego Olite, and Mireya Álvarez Toste. La contaminación del aire: su repercusión como problema de salud. *Revista Cubana de Higiene y Epidemiología*, 44:0 – 0, 08 2006.

[2] Francisco Vargas Marcos. La contaminación ambiental como factor determinante de la salud. *Esp Salud Pública*, 79:117 – 127, 2005.

[3] J. S. Serna-Trejos, E. Agudelo-Quintero, and S. G. Bermúdez-Moyano. Machine Learning en ciencias de la salud: Usos y aplicaciones: Machine Learning in health sciences: Uses and applications. *Peruvian Journal of Health Care and Global Healthc*, 6:95 – 96, 2022.

[4] Marzyeh Ghassemi, Tristan Naumann, Peter Schulam, Andrew L Beam, Irene Y Chen, and Rajesh Ranganath. A review of challenges and opportunities in Machine Learning for Health. *AMIA Jt Summits Transl Sci Proc*, pages 191 – 200, 2020.

[5] Pablo Fernández-Navarro, Javier García-Pérez, Rebeca Ramis, Elena Boldo, and Gonzalo López-Abente. Industrial pollution and cancer in spain: An important public health issue. *Environmental Research*, 159:555–563, 2017.

[6] International Agency for Research on Cancer. IARC Monographs on the Identification of Carcinogenic Hazards to Humans. IARC Newsletter, 2024. Disponible en:
urlhttps://monographs.iarc.who.int/agents-classified-by-the-iarc/.

[7] Ana M. Vicedo Cabrera. Estudios sobre contaminación atmosférica y salud en el ámbito del consorcio mcc: resultados para el ozono. *Rev. salud ambient.*, 19(Espec. Congr.):64–120, 2019.

[8] Javier García-Pérez, Elena Boldo, Rebeca Ramis, Marina Pollán, Beatriz Pérez-Gómez, Nuria Aragonés, and Gonzalo López-Abente. Description of industrial pollution in spain. *BMC Public Health*, 7:40, 2007.

[9] Zezhi Peng, Bin Zhang, Diwei Wang, Xinyi Niu, Jian Sun, Hongmei Xu, Junji Cao, and Zhenxing Shen. Application of machine learning in atmospheric pollution research: A state-of-art review. *Science of The Total Environment*, 910:168588, 2024.

[10] Abdul-Lateef Balogun, Abdulwaheed Tella, Lavania Baloo, and Naheem Adebisi. A review of the inter-correlation of climate change, air pollution and urban sustainability using novel machine learning algorithms and spatial information science. *Urban Climate*, 40:100989, 2021.

[11] Mario Villatoro, Carlos Henríquez, and Freddy Sancho. Comparación de los interpoladores idw y kriging en la variación espacial de ph, ca, cice y p del suelo. *Agronomía Costarricense*, 32(1):95–105, 2008.

[12] José Alberto Rosales-Castillo, Víctor Manuel Torres-Meza, Gustavo Olaiz-Fernández, and Víctor H. Borja-Aburto. Los efectos agudos de la contaminación del aire en la salud de la población: evidencias de estudios epidemiológicos. *Salud Pública de México*, 43(6):544–555, 2001.

[13] YF Xing, YH Xu, MH Shi, and YX Lian. The impact of pm2.5 on the human respiratory system. *Journal of Thoracic Disease*, 8(1):E69–E74, January 2016.

[14] Philip J Landrigan. Air pollution and health. *The Lancet Public Health*, 2(1):e4–e5, 2017.

[15] J. Jason West, Aaron Cohen, Frank Dentener, Bert Brunekreef, Tong Zhu, Ben Armstrong, Michelle L. Bell, Michael Brauer, Gregory Carmichael, Dan L. Costa, Douglas W. Dockery, Michael Kleeman, Michal Krzyzanowski, Nino Künzli, Catherine Liousse, Shih-Chun Candice Lung, Randall V. Martin, Ulrich Pöschl, C. Arden III Pope, James M. Roberts, Armistead G. Russell, and Christine Wiedinmyer. What we breathe impacts our health: Improving understanding of the link between air pollution and health. *Environmental Science & Technology*, 50(10):4895–4904, May 2016.

[16] Gregor Kiesewetter, Wolfgang Schoepp, Chris Heyes, and Markus Amann. Modelling pm2.5 impact indicators in europe: Health effects and legal compliance. *Environmental Modelling Software*, 74:201–211, 2015.

[17] Junta de Andalucía, Unión Europea, and OSMAN. Calidad del aire interior. PDF, 2011. Depósito Legal GR 2672-2011 ISBN 978-84-964-5934-8.

[18] Neeta Kulkarni and Jonathan Grigg. Effect of air pollution on children. *Paediatrics and Child Health*, 18(5):238–243, 2008.

[19] Organización Mundial de la Salud. Contaminación del aire ambiente (exterior) y salud. Página web, 2024. Consultado el 27 de octubre de 2024.

[20] Ana Esplugues, Rosalía Fernández-Patier, Inma Aguilera, Carmen Iñíguez, Saúl García Dos Santos, Amelia Aguirre Alfaró, Marina Lacasaña, Marisa Estarlich, Joan O. Grimalte, Marieta Fernández, Marisa Rebagliato, María Salah, Adonina Tardón, Maties Torrent, María Dolores Martínez, Núria Ribas-Fitó, Jordi Sunyer, and Ferran Ballester. Exposición a contaminantes atmosféricos durante el embarazo y desarrollo prenatal y neonatal: protocolo de investigación en el proyecto inma (infancia y medio ambiente). *Gaceta Sanitaria*, 21(6):455–463, 2007.

[21] Radim J Sram, Blanka Binková, Jan Dejmek, and Martin Bobak. Ambient air pollution and pregnancy outcomes: A review of the literature. *Environmental Health Perspectives*, 113(4):375–382, 2005.

[22] Insa Korten, Kathryn Ramsey, and Philipp Latzin. Air pollution during pregnancy and lung development in the child. *Paediatric Respiratory Reviews*, 21:38–46, 2017.

[23] D.V. Bates. The effects of air pollution on children. *Environmental Health Perspectives*, 103(suppl 6):49–53, 1995.

[24] W. James Gauderman, Edward Avol, Frank Gilliland, Hita Vora, Duncan Thomas, Kiros Berhane, Rob McConnell, Nino Kuenzli, Fred Lurmann, Edward Rappaport, Helene Margolis, David Bates, and John Peters. The effect of air pollution on lung development from 10 to 18 years of age. *New England Journal of Medicine*, 351(11):1057–1067, 2004.

[25] Rosalba Rojas-Martinez, Rogelio Perez-Padilla, Gustavo Olaiz-Fernandez, Laura Mendoza-Alvarado, Hortensia Moreno-Macias, Teresa Fortoul, William McDonnell, Dana Loomis, and Isabelle Romieu. Lung function growth in children with long-term exposure to air pollutants in mexico city. *American Journal of Respiratory and Critical Care Medicine*, 176(4):377–384, August 2007.

[26] Francesca Dominici, Roger D Peng, Michelle L Bell, Luu Pham, Aidan McDermott, Scott L Zeger, and Jonathan M Samet. Fine particulate air pollution and hospital admission for cardiovascular and respiratory diseases. *JAMA*, 295(10):1127–1134, March 2006.

[27] Marzia Simoni, Sandra Baldacci, Sara Maio, Sonia Cerrai, Giuseppe Sarno, and Giovanni Viegi. Adverse effects of outdoor pollution in the elderly. *Journal of Thoracic Disease*, January 2015.

[28] J. Sunyer, M. Sáez, C. Murillo, and et al. Air pollution and emergency room admissions for chronic obstructive pulmonary disease: a 5-year study. *Am J Epidemiol*, 137:701–705, 1993.

[29] T. Götschi, J. Heinrich, J. Sunyer, and N. Künzli. Long-term effects of ambient air pollution on lung function: a review. *Epidemiology*, 19(5):690–701, 2008.

[30] Miguel Ángel Ceballos, Paco Segura, Eduardo Gutiérrez, Juan Carlos Gracia, Paco Ramos, Mariano Reaño, Bernardo García, Marta Orihuel, Miguel Ángel Ceballos, Dídac Navarro, Helena Prima, Carlos Arribas, Carlos Garrón, Xosé Veiras, Juan Bárcena, Pedro Belmonte, Eduardo Navascués, Francisco García, Pedro Luis Mier, Koldo Hernández, and Pablo Muñoz. La calidad del aire en el estado español durante 2023, junio 2024.

[31] Cristina Ortiz, Cristina Linares, Rocio Carmona, and Julio Díaz. Evaluation of short-term mortality attributable to particulate matter pollution in spain. *Environmental Pollution*, 224:541–551, 2017.

[32] Ana Ayuso-Álvarez, Javier García-Pérez, José-Matías Triviño-Juárez, Unai Larrinaga-Torrontegui, Mario González-Sánchez, Rebeca Ramis, Elena Boldo, Gonzalo López-Abente, Iñaki Galán, and Pablo Fernández-Navarro. Association between proximity to industrial chemical installations and cancer mortality in spain. *Environmental Pollution*, 260:113869, 2020.

[33] Xavier Querol, Jordi Massagué, Andrés Alastuey, Teresa Moreno, Gotzon Gangoiti, Enrique Mantilla, José Jaime Duéguez, Miguel Escudero, Eliseo Monfort, Carlos Pérez García-Pando, Hervé Petetin, Oriol Jorba, Víctor Vázquez, Jesús de la Rosa, Alberto Campos, Marta Muñóz, Silvia Monge, María Hervás, Rebeca Javato, and María J. Cornide. Lessons from the covid-19 air pollution decrease in spain: Now what? *Science of The Total Environment*, 779:146380, 2021.

[34] Javier Cárcel-Carrasco, Manuel Pascual-Guillamón, and Jaime Langa-Sanchis. Analysis of the effect of covid-19 on air pollution: perspective of the spanish

case. *Environmental Science and Pollution Research*, 28(27):36880–36893, jul 2021.

[35] Andrés Monzón and Maria-José Guerrero. Valuation of social and health effects of transport-related air pollution in madrid (spain). *Science of The Total Environment*, 334-335:427–434, 2004. Highway and Urban Pollution.

[36] World Health Organization. Ambient air pollution data portal, 2024. Available at: https://www.who.int/data/gho/data/themes/air-pollution/ambient-air-pollution (Accessed: 2024-12-05).

[37] World Health Organization. Ambient air pollution attributable death rate (per 100 000 population, age-standardized), 2024. Available at: https://www.who.int/data/gho/data/indicators/indicator-details/GHO/ambient-air-pollution-attributable-death-rate-(per-100-000-populati (Accessed: 2024-12-05).

[38] Deepak Kumar Rathore and Praveen Kumar Mannepalli. A review of machine learning techniques and applications for health care. pages 4–8, 2021.

[39] Miroslav Kubat, Ivan Bratko, and Ryszard Michalski. A review of machine learning methods. 1996.

[40] Ramzi Farhat, Yosra Mourali, Mohamed Jemni, and Houcine Ezzedine. An overview of machine learning technologies and their use in e-learning. pages 1–4, 2020.

[41] Thomas Rincy N and Roopam Gupta. A survey on machine learning approaches and its techniques:. pages 1–6, 2020.

[42] Susmita Ray. A quick review of machine learning algorithms. pages 35–39, 2019.

[43] Neha Sharma, Reecha Sharma, and Neeru Jindal. Machine learning and deep learning applications-a vision. *Global Transitions Proceedings*, 2(1):24–28, 2021. 1st International Conference on Advances in Information, Computing and Trends in Data Engineering (AICDE - 2020).

[44] Suja Cherukullapurath Mana, G. Kalaiarasi, Yogitha R, L Suji Helen, and R. Senthamil Selvi. Application of machine learning in healthcare: An analysis. pages 1611–1615, 2022.

[45] Kavitha S, Varuna S, and Ramya R. A comparative analysis on linear regression and support vector regression. pages 1–5, 2016.

[46] Rong Shen and Bao-wen Zhang. The research of regression model in machine learning field. *MATEC Web of Conferences*, 176:01033, 2018.

[47] Arys Carrasquilla-Batista, Alfonso Chacón-Rodríguez, Kattia Núñez-Montero, Olman Gómez-Espinoza, Johnny Valverde, and Maritza Guerrero-Barrantes. Regresión lineal simple y múltiple: aplicación en la predicción de variables naturales relacionadas con el crecimiento microalgal. *Revista Tecnología en Marcha*, 29(Suppl. 5):33–45, 2016. Accessed: 2025-01-20.

[48] Eva Ostertagová. Modelling using polynomial regression. *Procedia Engineering*, 48:500–506, 2012. Modelling of Mechanical and Mechatronics Systems.

[49] J. M. Astorga Gómez. Aplicación de modelos de regresión lineal para determinar las armónicas de tensión y corriente. *Ingeniería Energética*, XXXV(3):234–241, 2014.

[50] G. E. Chanchí Golondrino, W. Y. Campo Muñoz, and L. M. Sierra Martínez. Aplicación de la regresión polinomial para la caracterización de la curva del covid-19, mediante técnicas de machine learning. *Investigación e Innovación en Ingenierías*, 8(2):87–105, jul 2020.

[51] Soo-Jin Kim, Seung-Jong Bae, and Min-Won Jang. Linear regression machine learning algorithms for estimating reference evapotranspiration using limited climate data. *Sustainability*, 14(18), 2022.

[52] Philip J. Drew and John R.T. Monson. Artificial neural networks. *Surgery*, 127(1):3–11, 2000.

[53] Pranjali Satish Deshmukh. Travel time prediction using neural networks: A literature review. pages 1–5, 2018.

[54] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Copyright © 2024. All rights reserved., 2024. Draft of January 12, 2025.

[55] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd edition, 2009. Corrected 12th printing Jan 2017.

[56] Carlos Arana. Redes neuronales recurrentes: Análisis de los modelos especializados en datos secuenciales. Technical Report 797, Universidad del CEMA, Buenos Aires, Argentina, June 2021.

[57] Alice Gao. Lecture 7: Decision trees, November 2021. CS 486/686 Lecture 7.

[58] Feng-Jen Yang. An extended idea about decision trees. pages 349–354, 2019.

[59] Dan Roth. Decision trees. Sep 8, 2016, 2016. CS 446 Machine Learning Fall 2016, Scribe: Ben Zhou, C. Cervantes.

[60] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

[61] Bao Chong. K-means clustering algorithm: a brief review. *Academic Journal of Computing & Information Science*, 4(5):37–40, 2021.

[62] Ankita Dubey and Abha Choubey. A systematic review on k-means clustering techniques. *International Journal of Scientific Research Engineering & Technology (IJSRET)*, 6(6), June 2017.

[63] Alonso. Capítulo 6: Interpolación a partir de puntos e isolineas. Apuntes de clase.

[64] Vanesa Naranjo Jiménez. *Métodos de Interpolación para la Reconstrucción del Campo de Velocidad de Tráfico Vehicular*. Proyecto fin de carrera, Universidad Rey Juan Carlos, Escuela Técnica Superior de Ingeniería de Telecomunicación, 2010/2011.

[65] Universidad de Sevilla. Geometría computacional tema 3: Diagramas de voronoi, 2025. Último acceso: 5 de marzo de 2025.

[66] Tom Bobach. *Natural Neighbor Interpolation - Critical Assessment and New Contributions*. Dissertation (dr. rer. nat.), Technische Universität Kaiserslautern, Kaiserslautern, Germany, April 2008.

[67] Esri. *How IDW Works - ArcGIS Pro Documentation*. Environmental Systems Research Institute (Esri), 2024. Disponible en ArcGIS Pro Help.

[68] Wikipedia contributors. Inverse distance weighting, 2024. Wikipedia, The Free Encyclopedia. Consultado el 5 de marzo de 2025.

[69] Wojciech Maleika. Inverse distance weighting method optimization in the process of digital terrain model creation based on data collected from a multibeam echosounder. *Journal of Marine Science and Engineering*, Accepted, March 2020.

[70] Timo Denk. Cubic spline interpolation, 2017. Accessed: 18 March 2025.

[71] W.C.M. van Beers and J.P.C. Kleijnen. Kriging interpolation in simulation: a survey. In *Proceedings of the 2004 Winter Simulation Conference, 2004.*, volume 1, page 121, 2004.

[72] Ramón Giraldo Henao. *Introducción a la Geoestadística: Teoría y Aplicación.* Universidad Nacional de Colombia, Bogotá, Colombia, 2021.

[73] Vladimir M. Krasnopolsky. Neural network emulations for complex multidimensional geophysical mappings: Applications of neural network techniques to atmospheric and oceanic satellite retrievals and numerical modeling. *Reviews of Geophysics*, 45(4):RG3009, 2007. Received 19 April 2006; revised 23 January 2007; accepted 19 March 2007; published 25 September 2007.

[74] Claudio García Vargas. Exterior differential systems: Frobenius theorem. Master's thesis, Departamento de Análisis Matemático, Universidad de La Laguna, La Laguna, Tenerife, 2016.

[75] H. Hernández. Redes neuronales vs kriging, July 12 2023. Accessed: 2025-03-19.

[76] Raúl E. López Briega. Distribuciones de probabilidad con python. `https://relopezbriega.github.io/blog/2016/06/29/distribuciones-de-probabilidad-con-python/`, 2016. Blog post en Matemáticas, análisis de datos y Python. Licencia BSD.

[77] Agata Gładka, Joanna Rymaszewska, and Tomasz Zatoński. Impact of air pollution on depression and suicide. *International journal of occupational medicine and environmental health*, 31(6):711–721, 2018.

A. Código

A.1. Introducción

En los próximos códigos se realiza la carga de datos *.dat* con concentraciones de contaminantes. A continuación, se realiza la limpieza de los valores inválidos y filtrado de los válidos. Por último, se generaron histogramas con curvas de densidad para visualizar la distribución de cada contaminante y se representaron geográficamente las ubicaciones de las muestras en el mapa.

```
carpeta <- "Directorio de la carpeta"
archivos_dat <- list.files(carpeta, pattern = "\\.dat$", full.names =
    TRUE)

# Leer todos los archivos .dat y almacenarlos en variables con sus
    nombres
for (archivo in archivos_dat) {
  nombre_base <- tools::file_path_sans_ext(basename(archivo))
  df <- read.table(archivo, header = FALSE, sep = "", stringsAsFactors
      = FALSE)
  colnames(df) <- c("Longitud", "Latitud", "CONCENTRACIN")
  assign(nombre_base, df, envir = .GlobalEnv)
}

# Renombrar variables
NO2 <- Integrado_por_poblacion_mapa_final_NO2_bias_anual_blank_new
PM10 <- Integrado_por_poblacion_mapa_final_PM10_bias_anual_blank_new
PM25 <- Integrado_por_poblacion_mapa_final_PM25_bias_anual_blank_new
SO2 <- Integrado_por_poblacion_mapa_final_SO2_bias_anual_blank_new
CO <- `conc-anual-CO_mod_maxoctoh_2019_blank`
O3_1 <- mapa_diario_O3_26th_2019_nueva_metodologia_blank
O3_26 <- mapa_O3_1th_2019_nueva_metodologia_blank
```

Listing 33: Almacenamiento de parámetros contaminantes.

```
library(ggplot2)
library(maps)

png("Directorio.png", width = 800, height = 600)
ggplot() +
  borders("world", colour = "gray70", fill = "gray90") +
  geom_point(data = CO, aes(x = Longitud, y = Latitud), color = "red",
```

```
      alpha = 0.6) +
  coord_cartesian(xlim = c(-10,5),
                  ylim = c(30, 50)) +
  theme_minimal() +
  labs(title = "Ubicaciones de Puntos de CO", x = "Longitud", y = "
      Latitud")
dev.off()

summary(CO$Latitud)
```

Listing 34: Visualización de los datos en el mapa.

```
# Filtrar valores 1.70141E+038 -> NaN
CO$CONCENTRACIN[CO$CONCENTRACIN == 1.70141E+038] <- NaN
NO2$CONCENTRACIN[NO2$CONCENTRACIN == 1.70141E+038] <- NaN
O3_1$CONCENTRACIN[O3_1$CONCENTRACIN == 1.70141E+038] <- NaN
O3_26$CONCENTRACIN[O3_26$CONCENTRACIN == 1.70141E+038] <- NaN
PM10$CONCENTRACIN[PM10$CONCENTRACIN == 1.70141E+038] <- NaN
PM25$CONCENTRACIN[PM25$CONCENTRACIN == 1.70141E+038] <- NaN
SO2$CONCENTRACIN[SO2$CONCENTRACIN == 1.70141E+038] <- NaN

# Filtrar las filas donde CONCENTRACION es mayor o igual a 0 y no es NA
    -> valores validos de CO
VV_CO <- CO[CO$CONCENTRACIN >= 0 & !is.na(CO$CONCENTRACIN), ]
VV_NO2 <- NO2[NO2$CONCENTRACIN >= 0 & !is.na(NO2$CONCENTRACIN), ]
VV_O3_1 <- O3_1[O3_1$CONCENTRACIN >= 0 & !is.na(O3_1$CONCENTRACIN),
    ]
VV_O3_26 <- O3_26[O3_26$CONCENTRACIN >= 0 & !is.na(O3_26$
    CONCENTRACIN), ]
VV_PM10 <- PM10[PM10$CONCENTRACIN >= 0 & !is.na(PM10$CONCENTRACIN),
    ]
VV_PM25 <- PM25[PM25$CONCENTRACIN >= 0 & !is.na(PM25$CONCENTRACIN),
    ]
VV_SO2 <- SO2[SO2$CONCENTRACIN >= 0 & !is.na(SO2$CONCENTRACIN), ]
```

Listing 35: Eliminacion de filas vacías y filtrado de datos.

```
png("Directorio\\histo_CO.png", width = 800, height = 600)
datos_filtrados <- VV_CO$CONCENTRACIN[VV_CO$CONCENTRACIN >= 0 & VV_
    CO$CONCENTRACIN <= 1]
hist(datos_filtrados, # Crear histograma con esos valores
     main = "Concentraci n CO",
     xlab = "Concentraci n",
```

```
      col = "lightblue",
      border = "black",
      probability = TRUE,
      xlim = c(0, 1),
      breaks = seq(0, 1, by = 0.01))
lines(density(datos_filtrados), col = "red", lwd = 2) #Curva de
    densidad
dev.off()

png("Directorio.png", width = 800, height = 600)
datos_filtrados <- VV_NO2$CONCENTRACIN[VV_NO2$CONCENTRACIN >= 0 &
    VV_NO2$CONCENTRACIN <= 25]
hist(datos_filtrados,
      main = "Concentraci n NO2",
      xlab = "Concentraci n",
      col = "lightblue",
      border = "black",
      probability = TRUE,
      xlim = c(0, 25),
      breaks = seq(0, 25, by = 0.5))
lines(density(datos_filtrados), col = "red", lwd = 2)

dev.off()

png("Directorio.png", width = 800, height = 600)
datos_filtrados <- VV_O3_1$CONCENTRACIN[VV_O3_1$CONCENTRACIN >= 80
    & VV_O3_1$CONCENTRACIN <= 140]
hist(datos_filtrados,
      main = "Concentraci n O3_1",
      xlab = "Concentraci n",
      col = "lightblue",
      border = "black",
      probability = TRUE,
      xlim = c(80, 140),
      breaks = seq(80, 140, by = 2))
lines(density(datos_filtrados), col = "red", lwd = 2)
dev.off()

png("Directorio.png", width = 800, height = 600)
datos_filtrados <- VV_O3_26$CONCENTRACIN[VV_O3_26$CONCENTRACIN >=
    90 & VV_O3_26$CONCENTRACIN <= 250]
hist(datos_filtrados,
```

```
      main = "Concentraci n O3_26",
      xlab = "Concentraci n",
      col = "lightblue",
      border = "black",
      probability = TRUE,
      xlim = c(90, 250),
      breaks = seq(90, 250, by = 2))
lines(density(VV_O3_26$CONCENTRACIN), col = "red", lwd = 2)
dev.off()

png("Directorio.png", width = 800, height = 600)
datos_filtrados <- VV_PM10$CONCENTRACIN[VV_PM10$CONCENTRACIN >= 5 &
    VV_PM10$CONCENTRACIN <= 30]
hist(datos_filtrados,
      main = "Concentraci n PM10",
      xlab = "Concentraci n",
      col = "lightblue",
      border = "black",
      probability = TRUE,
      xlim = c(5, 30),
      breaks = seq(5, 30, by = 1))
lines(density(VV_PM10$CONCENTRACIN), col = "red", lwd = 2)
dev.off()

png("Directorio.png", width = 800, height = 600)
datos_filtrados <- VV_PM25$CONCENTRACIN[VV_PM25$CONCENTRACIN >= 0 &
    VV_PM25$CONCENTRACIN <= 15]
hist(datos_filtrados,
      main = "Concentraci n PM_25",
      xlab = "Concentraci n",
      col = "lightblue",
      border = "black",
      probability = TRUE,
      xlim = c(0, 15),
      breaks = seq(0, 15, by = 0.2))
lines(density(datos_filtrados), col = "red", lwd = 2)
dev.off()

png("Directorio.png", width = 800, height = 600)
datos_filtrados <- VV_SO2$CONCENTRACIN[VV_SO2$CONCENTRACIN >= 1 &
   VV_SO2$CONCENTRACIN <= 5]
hist(datos_filtrados,
```

```
728       main = "Concentraci n SO2",
          xlab = "Concentraci n",
730       col = "lightblue",
          border = "black",
732       probability = TRUE,
          xlim = c(1, 5),
734       breaks = seq(1, 5, by = 0.05))
lines(density(VV_SO2$CONCENTRACIN), col = "red", lwd = 2)
736 dev.off()


738
png("Directorio.png", width = 800, height = 600)
740 ggplot() +
  borders("world", colour = "gray70", fill = "gray90") +
742   geom_point(data = VV_CO, aes(x = Longitud, y = Latitud), color = "red
      ", alpha = 0.6) +
  coord_cartesian(xlim = c(-10.35,5.35),
744                   ylim = c(34.95, 44.8)) +
  theme_minimal() +
746   labs(title = "Ubicaciones de Puntos de C O ", x = "Longitud", y = "
      Latitud")
dev.off()
```

Listing 36: Creación de histogramas para cada contaminante y nube de puntos tras la limpieza de variables.

A.2. Interpolación

A.2.1. Vecino más próximo

Este código implementa el método de kNN (*k-Nearest Neighbors*) con el fin de estimar las concentraciones de CO en función de la ubicación geográfica. Evalúa el rendimiento variando el número de vecinos (k) y analizando los errores absoluto, relativo y tiempo de ejecución para cada valor. Posterior a esto se evalúa el resultado en gráficos comparativos.

```
748 tiempo <- system.time({
library(class)
750 library(FNN)


752 # Numero de vecinos cercanos
```

```
k <- 6
n <- nrow(VV_CO)

# Creacion de variables vacias con el fin de obtener datos en el futuro
predicciones <- numeric(length = nrow(VV_CO))
error <- numeric(length = nrow(VV_CO))
error_relativo <- numeric(length = nrow(VV_CO))
X <- VV_CO[, c("Longitud", "Latitud")]

for (i in 1:n) {
  # Punto de prueba
  punto_test <- X[i, , drop = FALSE]

  # Puntos de entrenamiento
  X_train <- X[-i, , drop = FALSE]
  y_train <- VV_CO$CONCENTRACIN[-i]

  # Aplicar el metodo
  prediccion <- knn.reg(train = X_train, test = punto_test, y = y_train
      , k = k)

  predicciones[i] <- prediccion$pred
  error[i] <- abs(predicciones[i] - VV_CO$CONCENTRACIN[i])
  error_relativo[i] <- error[i] / VV_CO$CONCENTRACIN[i] * 100
}

# Imprimir las predicciones y errores
data.frame(ID = 1:nrow(VV_CO), Real = VV_CO$CONCENTRACIN, Prediccion
    = predicciones, Error = error, ErrorRel = error_relativo)

# Calcular el promedio de error
promedio_error <- mean(error)
error_relativo_promedio <- mean(error_relativo, na.rm = TRUE)
})

# REPRESENTACION DE LOS DATOS
E <- numeric(length = k)
E_r <- numeric(length = k)
T <- numeric(length = k)
E <- c(0.01857866, 0.01174832, 0.008681394,0.007231609,
    0.008878408,0.009705822)
E_r <- c(5.437721,3.424426, 2.46592,1.996351, 2.491139, 2.760753 )
```

```
792 T <- c(266.78, 258.26, 255.12,235.25, 238.59, 235.55)
    nombres <- c("k=1", "k=2", "k=3", "k=4", "k=5", "k=6")
794
    # Graficar Error Promedio
796 png("Directorio.png", width = 800, height = 600)
    plot(1:k, E, type = "b", col = "blue", pch = 19,lwd = 2, xlab = "Casos"
        , ylab = "Error",
798       xaxt = "n", main = "Error",cex.axis = 1.5, cex.lab = 1.5, cex.main
            = 2)
    axis(1, at = 1:k, labels = nombres)
800 dev.off()

802 # Graficar Error Relativo
    png("Directorio.png", width = 800, height = 600)
804 plot(1:k, E_r, type = "b", col = "red", pch = 19,lwd = 2, xlab = "Casos
    ", ylab = "Error Relativo",
       xaxt = "n", main = "Error Relativo",cex.axis = 1.5, cex.lab = 1.5,
            cex.main = 2)
806 axis(1, at = 1:k, labels = nombres)
    dev.off()
808
    # Graficar Tiempo
810 png("Directorio.png", width = 800, height = 600)
    plot(1:k, T, type = "b", col = "orange", pch = 19,lwd = 2, xlab = "
    Casos", ylab = "Tiempo",
812       xaxt = "n", main = "Tiempo",cex.axis = 1.5, cex.lab = 1.5, cex.
            main = 2)
    axis(1, at = 1:k, labels = nombres)
814 dev.off()
```

Listing 37: Aplicación del método kNN y determinación del error con diversos valores de K.

A.2.2. Kriging

Este *script* realiza la interpolación espacial de concentraciones de CO haciendo uso del método de Kriging. Se realizan ajustes de variogramas esféricos y exponenciales y se evalúa el método mediante validación cruzada. Posterior a esto se visualizan los resultados mediante mapas de calor, permitiendo así analizar la distribución geográfica del contaminante y precisión del ajuste.

```
816 library(sp)
    library(lattice)
818 library(gstat) # Libreria para hacer kriging y variogramas

820 # Conversion a un Spacial Data Frame (spdf)
    library(sp)
822 tiempo <- system.time({
    coords <- VV_CO[, c("Longitud", "Latitud")]
824 datos <- VV_CO["CONCENTRACIN"]
    puntos <- SpatialPoints(coords)
826 spdf <- SpatialPointsDataFrame(puntos, data = datos)

828 # Creacion de variogramas exponencial y esferico
    variogram_exp <- variogram(CONCENTRACIN ~ 1, spdf)
830 variograma <- variogram(CONCENTRACIN ~ 1, spdf, width = 0.3)
    ajuste_exp <- fit.variogram(variograma, model = vgm(psill = 60, "Exp",
        range = 200, nugget = 1))
832 ajuste_esf <- fit.variogram(variograma, model = vgm(psill = 60, "Sph",
        range = 200, nugget = 1))

834 png("Directorio.png", width = 8, height = 4, units = "in", res = 300)
    plot(variograma, ajuste_exp, main = "Ajuste Exponencial CO")
836 dev.off()

838 # Coeficiente de determinacion
    SSErr<-attr(ajuste_esf,"SSErr")
840 weig<-variograma$np / variograma$dist^2
    SStot<- sum(weig*(variograma$gamma - mean(variograma$gamma))^2)
842 (R2<-1-SSErr / SStot)

844 SSErr<-attr(ajuste_exp,"SSErr")
    weig<-variograma$np / variograma$dist^2
846 SStot<- sum(weig*(variograma$gamma - mean(variograma$gamma))^2)
    (R2<-1-SSErr / SStot)
848
    # Definir las coordenadas de la cuadricula
850 grd <- expand.grid(Longitud = seq(min(VV_CO$Longitud), max(VV_CO$
        Longitud), length.out = 100),
                         Latitud = seq(min(VV_CO$Latitud), max(VV_CO$Latitud)
                             , length.out = 100))
852 coordinates(grd) <- ~Longitud + Latitud
```

```
# Realizar la interpolacion con el modelo ajustado (esf rico en este
    caso)
ns.k <- gstat::krige(CONCENTRACIN ~ 1, spdf, grd, model = ajuste_esf)
interpolacion <- ns.k

set.seed(123)
spdf_sample <- spdf[sample(1:nrow(spdf), 1000), ]
cv_sample <- gstat::krige.cv(CONCENTRACIN ~ 1, spdf_sample, model =
    ajuste_exp, nfold = 1000)

# Calculo de error promedio y relativo
error_promedio <- abs(mean(cv_sample$residual))
error_relativo <- mean(abs(cv_sample$residual / cv_sample$observed), na
    .rm = TRUE) * 100  # En porcentaje

lower <- cv_sample$var1.pred - 1.96 * sqrt(cv_sample$var1.var)
upper <- cv_sample$var1.pred + 1.96 * sqrt(cv_sample$var1.var)

# Contar cuantos valores reales estan dentro del intervalo
dentro_intervalo <- cv_sample$observed >= lower & cv_sample$observed <=
     upper

# Porcentaje de confianza
porcentaje_confianza <- mean(dentro_intervalo, na.rm = TRUE) * 100

# Visualizacion de la interpolacion
spplot(interpolacion, "var1.pred", main = "Interpolaci n de la
    Concentraci n CO",
       col.regions = heat.colors(100),
       at = seq(min(interpolacion$var1.pred), max(interpolacion$var1.
           pred), length.out = 100))

png("Directorio.png", width = 8, height = 4, units = "in", res = 300)
levelplot(var1.pred ~ Longitud + Latitud, data = as.data.frame(
    interpolacion),
          main = "Interpolaci n de la Concentraci n CO",
          col.regions = heat.colors(100),
          xlim = c(min(VV_CO$Longitud) - 1, max(VV_CO$Longitud) + 1),
          ylim = c(min(VV_CO$Latitud) - 1, max(VV_CO$Latitud) + 1))
dev.off()

})
```

Listing 38: Interpolación Espacial de Concentraciones de CO mediante Kriging y Evaluación del Modelo.

A.2.3. IDW e IDW^6

Este código implementa el método de IDW (*Inverse Distance Weighting*) para estimar concentraciones de CO. Se aplicó la validación cruzada *leave-one-out* para evaluar la precisión del modelo. Posteriormente, se analizó el impacto del número de vecinos (k) sobre la calidad de las estimaciones. Finalmente se exploró el efecto de la potencia del inverso de la distancia sobre el rendimiento del modelo (IDW^p).

```
library(ggplot2)

tiempo <- system.time({

# Crear vectores para guardar resultados
errores_absolutos <- numeric(nrow(VV_CO))
errores_relativos <- numeric(nrow(VV_CO))
confianzas <- numeric(nrow(VV_CO))

for (i in 1:nrow(VV_CO)) {
  # Punto a predecir
  punto_real <- VV_CO[i, ]
  localizacion <- c(punto_real$Latitud, punto_real$Longitud)
  CO_restantes <- VV_CO[-i, ]

  # Calcular distancias a todos los otros puntos
  distancias <- sqrt((CO_restantes$Latitud - localizacion[1])^2 +
                     (CO_restantes$Longitud - localizacion[2])^2)

  # Parche: Evitar divisi n por 0
  distancias[distancias == 0] <- 1e-10

  # Seleccionar los k puntos m s cercanos
  k <- 4
  puntos_cercanos <- CO_restantes[order(distancias), ][1:k, ]
  distancias_k <- distancias[order(distancias)][1:k]

  # Pesos inversos a la distancia (IDW)
  pesos <- 1 / (distancias_k)
```

```
  pesos_normalizados <- pesos / sum(pesos)

  # Estimar concentracion
  concentraciones <- puntos_cercanos$CONCENTRACIN
  conc_estim <- sum(pesos_normalizados * concentraciones)

  # Calcular y almacenar errores
  error_abs <- abs(conc_estim - punto_real$CONCENTRACIN)
  error_rel <- error_abs / punto_real$CONCENTRACIN
  confianza <- (1 - error_rel) * 100
  errores_absolutos[i] <- error_abs
  errores_relativos[i] <- error_rel
  confianzas[i] <- confianza
}

# Calcular media de error absoluto y relativo
error_medio_absoluto <- mean(errores_absolutos)
error_medio_relativo <- mean(errores_relativos)


# ANALISIS DE FUNCIONAMIENTO DEL METODO EN RELACI N A "K"

# Crear un data.frame para almacenar los resultados por valor de k
resultados_k <- data.frame(
  k = integer(),
  error_absoluto_promedio = numeric(),
  error_relativo_promedio = numeric(),
  confianza_promedio = numeric()
)

# Bucle para probar diferentes valores de k (de 2 a 8)
for (k in 2:8) {

  errores_absolutos <- numeric(nrow(VV_CO))
  errores_relativos <- numeric(nrow(VV_CO))
  confianzas <- numeric(nrow(VV_CO))

  for (i in 1:nrow(VV_CO)) {
    punto_real <- VV_CO[i, ]
    localizacion <- c(punto_real$Latitud, punto_real$Longitud)
    CO_restantes <- VV_CO[-i, ]
    distancias <- sqrt((CO_restantes$Latitud - localizacion[1])^2 +
```

```
                        (CO_restantes$Longitud - localizacion[2])^2)
    distancias[distancias == 0] <- 1e-10
    puntos_cercanos <- CO_restantes[order(distancias), ][1:k, ]
    distancias_k <- distancias[order(distancias)][1:k]
    pesos <- 1 / (distancias_k^2)
    pesos_normalizados <- pesos / sum(pesos)
    concentraciones <- puntos_cercanos$CONCENTRACIN
    conc_estim <- sum(pesos_normalizados * concentraciones)
    error_abs <- abs(conc_estim - punto_real$CONCENTRACIN)
    error_rel <- error_abs / punto_real$CONCENTRACIN
    confianza <- (1 - error_rel) * 100
    errores_absolutos[i] <- error_abs
    errores_relativos[i] <- error_rel
    confianzas[i] <- confianza
  }

  # Calcular y almacenar promedios para este valor de k
  error_medio_absoluto <- mean(errores_absolutos)
  error_medio_relativo <- mean(errores_relativos)
  confianza_promedio <- mean(confianzas)
  resultados_k <- rbind(resultados_k, data.frame(
    k = k,
    error_absoluto_promedio = error_medio_absoluto,
    error_relativo_promedio = error_medio_relativo,
    confianza_promedio = confianza_promedio
  ))
}

# Grafico de error absoluto
png("Directorio.png", width = 8, height = 4, units = "in", res = 300)
grafico_error_abs <- ggplot(resultados_k, aes(x = k, y = error_absoluto
    _promedio)) +
  geom_line(color = "steelblue", size = 1.2) +
  geom_point(color = "steelblue", size = 3) +
  theme_minimal() +
  labs(title = "Error Absoluto Promedio vs. k",
       x = "N mero de vecinos (k)",
       y = "Error absoluto promedio")
print(grafico_error_abs)
dev.off()

# Grafico de error relativo
```

```
png("Directorio.png", width = 8, height = 4, units = "in", res = 300)
grafico_error_rel <- ggplot(resultados_k, aes(x = k, y = error_relativo
    _promedio * 100)) +
  geom_line(color = "firebrick", size = 1.2) +
  geom_point(color = "firebrick", size = 3) +
  theme_minimal() +
  labs(title = "Error Relativo Promedio vs. k",
       x = "N mero de vecinos (k)",
       y = "Error relativo promedio (%)")
print(grafico_error_rel)
dev.off()

# Almacenamiento de los datos de potencia
resultados_potencia <- data.frame(
  potencia = numeric(),
  error_absoluto_promedio = numeric(),
  error_relativo_promedio = numeric(),
  confianza_promedio = numeric()
)

# ANALISIS DEL MTODO EN RELACION A LA POTENCIA (IDW^p)
# Valores de potencia a evaluar (de 2 a 16)
potencias <- 2:16
k <- 4

for (p in potencias) {
  print(p)
  errores_absolutos <- numeric(nrow(VV_CO))
  errores_relativos <- numeric(nrow(VV_CO))
  confianzas <- numeric(nrow(VV_CO))

  for (i in 1:nrow(VV_CO)) {

    punto_real <- VV_CO[i, ]
    localizacion <- c(punto_real$Latitud, punto_real$Longitud)
    CO_restantes <- VV_CO[-i, ]

    distancias <- sqrt((CO_restantes$Latitud - localizacion[1])^2 +
                       (CO_restantes$Longitud - localizacion[2])^2)
    distancias[distancias == 0] <- 1e-10

    puntos_cercanos <- CO_restantes[order(distancias), ][1:k, ]
```

```
1042      distancias_k <- distancias[order(distancias)][1:k]

1044      pesos <- 1 / (distancias_k^p)
          pesos_normalizados <- pesos / sum(pesos)
1046
          concentraciones <- puntos_cercanos$CONCENTRACIN
1048      conc_estim <- sum(pesos_normalizados * concentraciones)

1050      error_abs <- abs(conc_estim - punto_real$CONCENTRACIN)
          error_rel <- error_abs / punto_real$CONCENTRACIN
1052      confianza <- (1 - error_rel) * 100

1054      errores_absolutos[i] <- error_abs
          errores_relativos[i] <- error_rel
1056      confianzas[i] <- confianza
        }
1058
        # Calcular promedios para esta potencia
1060    error_medio_absoluto <- mean(errores_absolutos)
        error_medio_relativo <- mean(errores_relativos)
1062    confianza_promedio <- mean(confianzas)

1064    # Guardar resultados
        resultados_potencia <- rbind(resultados_potencia, data.frame(
1066      potencia = p,
          error_absoluto_promedio = error_medio_absoluto,
1068      error_relativo_promedio = error_medio_relativo,
          confianza_promedio = confianza_promedio
1070    ))
}
1072
library(ggplot2)
1074
png("Directorio\\IDW_epromedio_potencia.png",
1076      width = 8, height = 4, units = "in", res = 300)
grafico_error_potencia <- ggplot(resultados_potencia, aes(x = potencia,
      y = error_absoluto_promedio)) +
1078  geom_line(color = "darkgreen", size = 1.2) +
    geom_point(color = "darkgreen", size = 3) +
1080  theme_minimal() +
    labs(
1082    title = "Error Absoluto Promedio vs. Potencia (IDW)",
```

```
    x = "Potencia",
    y = "Error absoluto promedio"
  )
print(grafico_error_potencia)
dev.off()
```

Listing 39: Evaluación de parámetros del método IDW para la estimación de concentraciones de CO

A.2.4. Árboles de decisión

Se implementó un modelo de árbol de decisión usando la librería *rpart* para la interpolación de contaminantes. La validación se realizó por el método *Leave-One-Out*. Se calcularon y almacenaron los errores absolutos y relativos y, finalmente se obtuvieron las métricas promedio de estos errores.

```
library(rpart)

tiempo <- system.time({
# Vectores para almacenar errores
errores_absolutos <- numeric(nrow(VV_CO))
errores_relativos <- numeric(nrow(VV_CO))

# Validacion cruzada Leave-One-Out
for (i in 1:nrow(VV_CO)) {

  punto_real <- VV_CO[i, ]
  datos_entrenamiento <- VV_CO[-i, ]

  # Modelo:
  modelo_cv <- rpart(CONCENTRACIN ~ Longitud + Latitud, data = datos_
      entrenamiento, method = "anova")
  prediccion <- predict(modelo_cv, newdata = punto_real)

  # Calcular errores
  error_abs <- abs(prediccion - punto_real$CONCENTRACIN)
  error_rel <- error_abs / punto_real$CONCENTRACIN
  errores_absolutos[i] <- error_abs
  errores_relativos[i] <- error_rel
}

error_medio_absoluto <- mean(errores_absolutos)
```

```
error_medio_relativo <- mean(errores_relativos)
})
```

Listing 40: Validación Cruzada Leave-One-Out con Árboles de Decisión para Predicción de Concentración de Contaminantes.

A.2.5. Bosques aleatorios

Este código entrena un modelo de *Random Forest* para estimar la concentración de contaminantes utilizando las coordenadas geográficas como variables predictoras. La evaluación se realizó mediante el método 10-fol. A partir de las predicciones cruzadas y los valores se calcularon los errores y se obtuvieron los promedios como métricas globales de desempeño del modelo.

```
library(caret)
library(randomForest)

tiempo <- system.time({
# Configurar validaci n cruzada 10-fold
set.seed(123)
control <- trainControl(method = "cv", number = 10, savePredictions = "
    final")
modelo_caret_rf <- train(CONCENTRACIN ~ Longitud + Latitud,
                         data = VV_CO,
                         method = "rf",
                         trControl = control,
                         ntree = 100)

# Obtener predicciones cruzadas y valores reales
predicciones <- modelo_caret_rf$pred$pred
observados <- modelo_caret_rf$pred$obs

# Calcular errores
errores_absolutos <- abs(predicciones - observados)
errores_relativos <- errores_absolutos / observados
error_medio_absoluto <- mean(errores_absolutos)
error_medio_relativo <- mean(errores_relativos) * 100  # en porcentaje
})
```

Listing 41: Validación Cruzada 10-Fold con Random Forest para Predicción de Concentración de Contaminantes.

A.3. Aplicación de la interpolación a la base de datos

A.3.1. Importación de las bases de datos

Se importaron y limpiaron datos de contaminantes ambientales y registros de salud, integrando coordenadas geográficas mediante códigos postales. Se realizó un análisis exploratorio para visualizar la distribución espacial y caracterizar la población según edad, sexo y prevalencia de enfermedades, estableciendo así las primeras impresiones sobre la base de datos.

```
carpeta_principal <- "Directorio"

# Obtener todos los archivos .DAT en todas las subcarpetas
archivos_dat <- list.files(carpeta_principal, pattern = "\\.dat$", full.names = TRUE, recursive = TRUE)

# Almacenar los datos importados
for (archivo in archivos_dat) {
  nombre_base <- tools::file_path_sans_ext(basename(archivo))
  df <- read.table(archivo, header = FALSE, sep = "", stringsAsFactors = FALSE)
  colnames(df) <- c("Longitud", "Latitud", "CONCENTRACIN")
  assign(nombre_base, df, envir = .GlobalEnv)
}
contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")

variables_datos <- ls(pattern = paste0("^[0-9]{4}_(", paste(contaminantes, collapse = "|"), ")$"))

for (var_name in variables_datos) {
  df <- get(var_name)

  # Reemplazar 1.70141E+038 por NaN
  df$CONCENTRACIN[df$CONCENTRACIN == 1.70141E+038] <- NaN

  # Filtrar valores no negativos
  df <- df[df$CONCENTRACIN >= 0 & !is.na(df$CONCENTRACIN), ]

  # Sobrescribir la variable original con los datos filtrados
  assign(var_name, df, envir = .GlobalEnv)
}
```

Listing 42: Importación, Limpieza y Análisis Exploratorio de Contaminación.

```
library(dplyr)
library(readxl)
library(ggplot2)
library(sf)

# Importacion datos salud
dat.salud <- read.csv("Directorio\\Datos_salud.csv", sep = ";", header
    = TRUE)
# Importacion de datos de centroides
Centroides.CP <- read_excel("Directorio\\CentroidesCP.xlsx")

CP <- Centroides.CP %>%
  select(COD_POSTAL, xcoord, ycoord)%>%
  rename(
    Longitud = xcoord,
    Latitud = ycoord
)

# Asegurar que COD_POSTAL en dat.salud y CP tengan 5 cifras
dat.salud$COD_POSTAL <- sprintf("%05d", as.integer(dat.salud$COD_POSTAL
    ))
CP$COD_POSTAL <- sprintf("%05d", as.integer(CP$COD_POSTAL))
CP <- CP[!duplicated(CP$COD_POSTAL), ]

# Realizar el merge entre CP.dat.salud y CP para agregar Longitud y
    Latitud a la base de datos de salud.
dat.salud <- merge(dat.salud, CP[, c("COD_POSTAL", "Longitud", "Latitud
    ")], by = "COD_POSTAL", all.x = TRUE)
codigos_no_encontrados <- dat.salud$COD_POSTAL[!dat.salud$COD_POSTAL %
    in% CP$COD_POSTAL]

# ANALISIS DE LOS DATOS DE SALUD
# Filtrar las filas con NA en Longitud y Latitud
dat.salud <- dat.salud %>%
  filter(!is.na(Longitud) & !is.na(Latitud))

# Convertir a objeto sf
dat.salud_sf <- st_as_sf(dat.salud, coords = c("Longitud", "Latitud"),
    crs = 4326)
```

```
34
png("Directorio.png", width = 800, height = 600)
36 ggplot(data = dat.salud_sf) +
  geom_sf(aes(color = COD_POSTAL), size = 2) +   # Graficar puntos
38   theme_minimal() +
  labs(title = "Puntos de toma de datos en el mapa",
40        x = "Longitud",
        y = "Latitud") +
42   theme(legend.position = "none")
dev.off()
44
# Cambio de formato de la fechas
46 fechas <- c(
  "Fec_Ini_Diab",
48   "Fec_Ini_Disl",
  "Fec_Ini_Hipot",
50   "Fec_Ini_Ob",
  "Fec_Ini_DepTab",
52   "Fec_Ini_TEA",
  "Fec_Ini_TAns",
54   "Fec_Ini_TCA",
  "Fec_Ini_ECV_Aguda",
56   "Fec_Ini_ECVMaldef",
  "Fec_Ini_Isq_Cer_trans",
58   "Fec_Ini_SECV",
  "Fec_Ini_ArtExt",
60   "Fec_Ini_ArtAbd",
  "Fec_Ini_Cisq",
62   "Fec_Ini_Hiperten",
  "Fec_Ini_IC",
64   "Fec_Ini_EstHep"
)
66 dat.salud[fechas] <- lapply(dat.salud[fechas], function(x) as.Date(as.
    character(x), format = "%Y%m%d"))
```

Listing 43: Importación, integración geográfica y preprocesamiento de datos de salud.

```
enfermedades <- c(
2   "Diabetes",
  "Dislipemia",
4   "Hipotiroidismo",
  "Obesidad",
6   "Dependenciatabaco",
```

```
  "Trastornoestadoanimo",
  "Trastornodeansiedad",
  "Trastornoconductaalimentaria",
  "ECVAguda",
  "ECVMaldef",
  "Isq_Cer_trans",
  "Secuelaenfermedadcerebrovascular",
  "Arteriopatideextremidades",
  "Arteriopatiintraabdominal",
  "Cardiopatiisquemica",
  "Hipertension",
  "Insuficienciacardiaca",
  "Esteatosishepatica"
)

total_pacientes <- nrow(dat.salud)
# Porcentaje de personas con cada enfermedad
porcentaje_enfermedades <- sapply(enfermedades, function(col) {
  sum(dat.salud[[col]] == 1, na.rm = TRUE) / total_pacientes * 100
})

sin_enfermedad <- rowSums(dat.salud[enfermedades] == 1, na.rm = TRUE)
    == 0
porcentaje_sin_enfermedad <- sum(sin_enfermedad) / total_pacientes *
    100
porcentajes <- c(porcentaje_enfermedades, Sin_Enfermedad = porcentaje_
    sin_enfermedad)
porcentajes <- round(porcentajes, 1)

# Graficas de datos sobre enfermedades
colores <- rainbow(length(porcentajes))
par(mar = c(4, 4, 4, 10))
pie(porcentajes,
    labels = paste0(round(porcentajes), "%"),
    main = "Distribuci n de enfermedades",
    col = colores,
    cex = 0.8)
legend("topright",
       inset = c(-0.35, 0),
       legend = names(porcentajes),
       fill = colores,
       xpd = TRUE,
```

```
        cex = 0.8,
        bty = "n")

png("Directorio.png", width = 800, height = 600)
hist(dat.salud$COD_EDAD,
      main = "Histograma de Edad",
      xlab = "Edad",
      ylab = "Frecuencia",
      col = "lightblue",
      border = "black"
      cex.main = 1.8,
      cex.lab = 1.5,
      cex.axis = 1.3)
dev.off()
s
sexo_freq <- table(dat.salud$COD_SEXO)

png("Directorio.png", width = 800, height = 600)
pie(sexo_freq,
    main = "Distribuci n por sexo",
    col = c("lightblue", "lightpink"),
    labels = paste0(names(sexo_freq), " (", round(100 * sexo_freq / sum
        (sexo_freq), 1), "%)"))
dev.off()
```

Listing 44: Preanalisis de los datos de salud.

A.3.2. Estudio optimización método *IDW*

En estos códigos se realizan dos interpolaciones por el método de *IDW* mediante dos formas, una manual, calculando las distancias a todos los puntos medidos y realizando el promedio, siendo este más lento parra el volumen de datos. El segundo bloque utiliza la función *get.knnx* que permite encontrar los vecinos más cercanos y calcular las predicciones de forma vectorizada, de esta manera se logra un rendimiento mayor sin necesidad de que disminuya la precisión, como lo confirma la comparación entre ambos métodos.

```
tiempo <- system.time({

  predicciones <- numeric(nrow(dat.salud))
for (i in 1:nrow(dat.salud)) {
```

```
  # Punto a predecir
  punto_real <- dat.salud[i, ]
  punto_real <- c(punto_real$Latitud, punto_real$Longitud)

  # Calcular distancias a todos los otros puntos
  distancias <- sqrt((`2019_CO`$Latitud - punto_real[1])^2 +
                     (`2019_CO`$Longitud - punto_real[2])^2)

  # Parche: evitar division por 0
  distancias[distancias == 0] <- 1e-10

  # Aplicacion del metodo IDW
  k <- 4
  puntos_cercanos <- `2019_CO`[order(distancias), ][1:k, ]
  distancias_k <- distancias[order(distancias)][1:k]
  pesos <- 1 / (distancias_k)
  pesos_normalizados <- pesos / sum(pesos)
  concentraciones <- puntos_cercanos$CONCENTRACIN
  conc_estim <- sum(pesos_normalizados * concentraciones)
  predicciones[i] <- conc_estim
}

predicciones
})
```

Listing 45: Interpolación espacial por método IDW, manual.

```
library(FNN)

dat.salud <- dat.salud[complete.cases(dat.salud[, c("Latitud", "
    Longitud")]), ]
coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])
tiempo <- system.time({
coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])
coords_CO <- as.matrix(`2019_CO`[, c("Latitud", "Longitud")])
k <- 4

# Aplicacion de la funcion get.knnx para rendimiento optimo
vecinos <- get.knnx(data = coords_CO, query = coords_salud, k = k)


distancias <- vecinos$nn.dist
indices <- vecinos$nn.index
```

```
distancias[distancias == 0] <- 1e-10
concentraciones_CO <- `2019_CO`$CONCENTRACIN
pesos <- 1 / distancias
pesos_normalizados <- pesos / rowSums(pesos)
concentraciones_vecinos <- matrix(concentraciones_CO[indices], ncol = k
    )
predicciones_f <- rowSums(pesos_normalizados * concentraciones_vecinos)
})
```

Listing 46: nterpolación espacial con IDW optimizada usando la función get.knnx.

```
# Diferencia absoluta entre ambas predicciones
max(abs(predicciones - predicciones_f))
# Diferencia relativa
mean(abs(predicciones - predicciones_f) / predicciones_f)
```

Listing 47: Comparación entre ambos tipos de predicciones.

A.3.3. Aplicación de la interpolación a toda la base de datos

Estos tres bloques de código realizan estimaciones de concentraciones mediante la interpolación espacial IDW e IDW^6. Con ambos métodos se genera predicciones en los puntos de la base de datos de salud.

```
library(FNN)

dat.salud <- dat.salud[complete.cases(dat.salud[, c("Latitud", "
    Longitud")]), ]
coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])

tiempo <- system.time({
# Definicion de parametros
anios <- 2008:2019
contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")
k <- 4

coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])

predicciones_total_idw <- data.frame(ID = seq_len(nrow(coords_salud)))

for (anio in anios) {
  for (cont in contaminantes) {
```

```
    nombre_var <- paste0(anio, "_", cont)

    if (exists(nombre_var)) {
      base <- get(nombre_var)

      if (all(c("Latitud", "Longitud", "CONCENTRACIN") %in% colnames(
          base))) {
        coords_contaminante <- as.matrix(base[, c("Latitud", "Longitud"
            )])
        concentraciones <- base$CONCENTRACIN

        vecinos <- get.knnx(data = coords_contaminante, query = coords_
            salud, k = k)
        distancias <- vecinos$nn.dist
        indices <- vecinos$nn.index

        distancias[distancias == 0] <- 1e-10

        # Metodo IDW
        pesos <- 1 / distancias
        pesos_normalizados <- pesos / rowSums(pesos)
        concentraciones_vecinos <- matrix(concentraciones[indices],
            ncol = k)

        predicciones <- rowSums(pesos_normalizados * concentraciones_
            vecinos)
        predicciones_total_idw[[nombre_var]] <- predicciones
      } else {
        warning(paste("Faltan columnas en", nombre_var))
      }
    } else {
      warning(paste("No existe la base de datos:", nombre_var))
    }
  }
}
})
```

Listing 48: Interpolación IDW con potencia 1

```
library(FNN)

dat.salud <- dat.salud[complete.cases(dat.salud[, c("Latitud", "
    Longitud")]), ]
```

```
coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])
tiempo <- system.time({
anios <- 2008:2019
contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")
k <- 4
coords_salud <- as.matrix(dat.salud[, c("Latitud", "Longitud")])
predicciones_total_idw6 <- data.frame(ID = seq_len(nrow(coords_salud)))
for (anio in anios) {
  for (cont in contaminantes) {
    nombre_var <- paste0(anio, "_", cont)
    if (exists(nombre_var)) {
      base <- get(nombre_var)
      if (all(c("Latitud", "Longitud", "CONCENTRACIN") %in% colnames(
          base))) {
        coords_contaminante <- as.matrix(base[, c("Latitud", "Longitud"
            )])
        concentraciones <- base$CONCENTRACIN
        vecinos <- get.knnx(data = coords_contaminante, query = coords_
            salud, k = k)
        distancias <- vecinos$nn.dist
        indices <- vecinos$nn.index
        distancias[distancias == 0] <- 1e-10

        # Pesos inversos a la distancia -> METODO IDW6
        pesos <- 1 / distancias^6
        pesos_normalizados <- pesos / rowSums(pesos)
        concentraciones_vecinos <- matrix(concentraciones[indices],
            ncol = k)
        predicciones <- rowSums(pesos_normalizados * concentraciones_
            vecinos)
        predicciones_total_idw6[[nombre_var]] <- predicciones
      } else {
        warning(paste("Faltan columnas en", nombre_var))
      }
    } else {
      warning(paste("No existe la base de datos:", nombre_var))
    }
  }
}
})
```

Listing 49: Interpolación IDW con potencia 6.)

```r
diff_abs <- abs(as.matrix(predicciones_total_idw) - as.matrix(
    predicciones_total_idw6))
mean_diff_rel <- mean(diff_abs / as.matrix(predicciones_total_idw6), na
    .rm = TRUE)
mean_diff_rel
```

Listing 50: Comparación de resultados entre IDW y IDW6.

A.3.4. Análisis temporal y exportación de concentraciones estimadas por IDW

Los siguientes bloques de código complementan el análisis de la interpolación espacial aplicada a contaminantes atmosféricos. Primero, se resumen las predicciones mediante la concentración media anual para cada contaminante, de esta manera, se puede visualizar tendencias. Posteriormente, se integran las predicciones con los datos de salud para crear una base de datos conjunta y poder exportar los datos en *Excel* y *CSV*.

```r
library(ggplot2)
medias_contaminantes <- data.frame()

# Bucle por anio y contaminante
for (anio in 2008:2019) {
  for (cont in c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")) {
    nombre_var <- paste0(anio, "_", cont)

    if (nombre_var %in% names(predicciones_total_idw6)) {
      media_valor <- mean(predicciones_total_idw6[[nombre_var]], na.rm
          = TRUE)
      medias_contaminantes <- rbind(medias_contaminantes, data.frame(
        Anio = anio,
        Contaminante = cont,
        Media = media_valor
      ))
    }
  }
}

# Crear subconjuntos segun agrupacion de rangos
```

```
grupo1 <- filter(medias_contaminantes, Contaminante %in% c("CO", "SO2",
    "PM25"))
grupo2 <- filter(medias_contaminantes, Contaminante %in% c("NO2", "PM10
    "))
grupo3 <- filter(medias_contaminantes, Contaminante %in% c("O3_1", "O3_
    26"))

# Graficas de resultados
graficar <- function(datos, titulo) {
  ggplot(datos, aes(x = Anio, y = Media, color = Contaminante)) +
    geom_line(size = 1.2) +
    geom_point(size = 3) +
    labs(
      title = titulo,
      x = "A o",
      y = "Concentraci n media estimada",
      color = "Contaminante"
    ) +
    theme_minimal() +
    theme(
      plot.title = element_text(hjust = 0.5, size = 18, face = "bold"),
      axis.title = element_text(size = 20),
      axis.text = element_text(size = 18),
      legend.title = element_text(size = 18),
      legend.text = element_text(size = 16),
      legend.position = "bottom"
    )
}

png("Directorio.png", width = 800, height = 600)
# Gr fica 1: contaminantes con valores peque os
graf1 <- graficar(grupo1, "Contaminantes con concentraci n baja (CO,
    SO2, PM25)")
print(graf1)
dev.off()

png("Directorio.png", width = 800, height = 600)
# Gr fica 2: contaminantes con concentraci n media
graf2 <- graficar(grupo2, "Contaminantes con concentraci n media (NO2,
    PM10)")
print(graf2)
dev.off()
```

```
1366
png("Directorio.png", width = 800, height = 600)
1368 # Gr fica 3: ozono con concentraci n alta
graf3 <- graficar(grupo3, "Contaminantes con concentraci n alta (O3_1,
    O3_26)")
1370 print(graf3)
dev.off()
```

Listing 51: Cálculo y visualización de medias anuales por contaminante.

```
library(writexl)
1374 # Union de bases de datos
dat.salud_idw <- cbind(dat.salud, predicciones_total_idw)
1376 dat.salud_idw6 <- cbind(dat.salud, predicciones_total_idw6)

1378 # Almacenamiento de datos en excel
write_xlsx(dat.salud_idw, "Directorio/dat_salud_idw.xlsx")
1380 write_xlsx(dat.salud_idw6, "Directorio/dat_salud_idw6.xlsx")

1382 # Almacenamiento de datos en CSV
write.csv(dat.salud_idw, "Directorio\\dat_salud_idw.csv", row.names =
    FALSE)
1384 write.csv(dat.salud_idw, "Directorio\\dat_salud_idw6.csv", row.names =
    FALSE)
```

Listing 52: Exportación de datos con predicciones IDW e IDW6.

A.4. Métodos de aprendizaje automático

A.4.1. Importación de los datos

Este código se encarga de cargar los archivos con las predicciones IDW para diferentes años y contaminantes, elimina las columnas de fechas y agrupa las columnas anuales de concentraciones de contaminante, de esta manera se simplifican los datos.

```
1386 # Importacion de la base de datos
dat_salud_idw<- read.csv("Directorio\dat_salud_idw.csv", header = TRUE,
    sep = ",")
1388 dat_salud_idw6<- read.csv("Directorio\\dat_salud_idw6.csv", header =
    TRUE, sep = ",")
```

```
# Eliminacion de las fechas de la base de datos
fechas <- c(
  "Fec_Ini_Diab",
  "Fec_Ini_Disl",
  "Fec_Ini_Hipot",
  "Fec_Ini_Ob",
  "Fec_Ini_DepTab",
  "Fec_Ini_TEA",
  "Fec_Ini_TAns",
  "Fec_Ini_TCA",
  "Fec_Ini_ECV_Aguda",
  "Fec_Ini_ECVMaldef",
  "Fec_Ini_Isq_Cer_trans",
  "Fec_Ini_SECV",
  "Fec_Ini_ArtExt",
  "Fec_Ini_ArtAbd",
  "Fec_Ini_Cisq",
  "Fec_Ini_Hiperten",
  "Fec_Ini_IC",
  "Fec_Ini_EstHep"
)
dat_salud_idw_mod <- dat_salud_idw[ , !(names(dat_salud_idw) %in%
    fechas)]

# Lista de contaminantes
contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")

# Parche: evitar que si ejecuto el codigo mas de una vez se rompan los
    datos
for (cont in contaminantes) {
  cols <- grep(paste0("^X\\d{4}_", cont, "$"), names(dat_salud_idw_mod)
      , value = TRUE)
  if (length(cols) > 0 && !(cont %in% names(dat_salud_idw_mod))) {

    # Calcular la media por fila ignorando NA
    dat_salud_idw_mod[[cont]] <- rowMeans(dat_salud_idw_mod[, cols], na
        .rm = TRUE)
    dat_salud_idw_mod <- dat_salud_idw_mod[, !(names(dat_salud_idw_mod)
        %in% cols)]
    cat(paste0("Contaminante ", cont, " procesado correctamente.\n"))
  } else if (cont %in% names(dat_salud_idw_mod)) {
    cat(paste0("Contaminante ", cont, " ya ha sido procesado
```

```
        previamente. No se repite.\n"))
  } else {
    cat(paste0("No se encontraron columnas anuales para el contaminante
        ", cont, ".\n"))
  }
}

# Eliminacion de parametros inutiles
dat_salud_idw_mod$COD_POSTAL <- NULL
dat_salud_idw_mod$DESC_PROVINCIA <- NULL
dat_salud_idw_mod$ID <- NULL
dat_salud_idw_mod$Longitud <- NULL
dat_salud_idw_mod$Latitud <- NULL
dat_salud_idw_mod$COD_CAP <- NULL
dat_salud_idw_mod$DESC_CAP <- NULL
dat_salud_idw_mod$DESC_DISTRITO <- NULL
dat_salud_idw_mod$COD_FINANCIACION <- NULL
dat_salud_idw_mod$COD_TRAMO_FARMACIA <- NULL
dat_salud_idw_mod$DESC_FINANCIACION <- NULL
dat_salud_idw_mod$NUM_PORC_TRAMO_FARMACIA <- NULL
```

Listing 53: Agrupación y cálculo de medias anuales por contaminante en los datos interpolados.

A.4.2. Árboles de decisión

El código presenta un conjunto de análisis mediante árboles de decisión para diferentes enfermedades: Diabetes, ECV mal definida, cardiopatías isquémicas, hipertensión e insuficiencia cardíaca. Para cada enfermedad, se transforma la variable objetivo en factor, se dividen los datos en entrenamiento y prueba. Se realizan pruebas de balanceo de datos para equilibrar las clases. Finalmente, se entrena el árbol de decisión y se genera la visualización de éste para evaluar el desempeño,

```
library(rpart)
library(rpart.plot)

# Asegurar la variable objetivo como factor
dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)
dat_salud_idw_mod$COD_DISTRITO <- as.factor(dat_salud_idw_mod$COD_
    DISTRITO)
dat_salud_idw_mod$COD_SEXO <- as.factor(dat_salud_idw_mod$COD_SEXO)
```

```
dat_salud_idw_mod$DESC_TRAMO_FARMACIA <- as.factor(dat_salud_idw_mod$
    DESC_TRAMO_FARMACIA)

# Dividir los datos en entrenamiento (80%) y prueba (20%)
set.seed(19354)
particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.8, ]
prueba <- dat_salud_idw_mod[particion >= 0.8, ]

# Crear el modelo del  rbol  de decisi n
modelo_Diabetes <- rpart(Diabetes ~ ., data = entrenamiento, method = "
    class")

# Graficar
png("Directorio.png", width = 1000, height = 800)
rpart.plot(modelo_Diabetes, cex = 2)
title(main = "  rbol  de Decisi n - Diabetes vs Contaminantes", cex.
    main = 2)
dev.off()

# Predecir en el conjunto de prueba
prediccion <- predict(modelo_Diabetes, prueba, type = "class")

# Matriz de confusi n
mc <- table(Real = prueba$Diabetes, Predicho = prediccion)
print(mc)

# Calcular la exactitud
exactitud <- sum(diag(mc)) / sum(mc)
cat("Exactitud en conjunto prueba:", round(exactitud * 100, 2), "%\n")
```

Listing 54: Modelo de árbol de decisión - diabetes - datos sin balancear

```
library(rpart)
library(rpart.plot)
library(dplyr)

dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)
dat_salud_idw_mod$COD_DISTRITO <- as.factor(dat_salud_idw_mod$COD_
    DISTRITO)
dat_salud_idw_mod$COD_SEXO <- as.factor(dat_salud_idw_mod$COD_SEXO)
dat_salud_idw_mod$DESC_TRAMO_FARMACIA <- as.factor(dat_salud_idw_mod$
    DESC_TRAMO_FARMACIA)
```

```
# Dividir los datos en entrenamiento (80%) y prueba (20%)
set.seed(19354)
library(caret)
trainIndex <- createDataPartition(dat_salud_idw_mod$Diabetes, p = 0.8,
    list = FALSE)
entrenamiento_original <- dat_salud_idw_mod[trainIndex, ]
prueba <- dat_salud_idw_mod[-trainIndex, ]

# Balanceo de datos
clase0 <- entrenamiento_original %>% filter(Diabetes == "0")
clase1 <- entrenamiento_original %>% filter(Diabetes == "1")
n_min <- min(nrow(clase0), nrow(clase1))
set.seed(123)
clase0_sub <- clase0 %>% sample_n(n_min)
clase1_sub <- clase1 %>% sample_n(n_min)
entrenamiento_balanceado <- bind_rows(clase0_sub, clase1_sub)

# Crear el modelo del arbol de decision
modelo_Diabetes <- rpart(Diabetes ~ ., data = entrenamiento_balanceado,
     method = "class")

# Graficar
png("Directorio.png", width = 1000, height = 800)
rpart.plot(modelo_Diabetes, cex = 2)
title(main = "  rbol  de Decisi n - Diabetes vs Contaminantes (
    Balanceado)", cex.main = 2)
dev.off()

# Predecir en el conjunto de prueba
predicci n <- predict(modelo_Diabetes, prueba, type = "class")

# Matriz de confusi n
mc <- table(Real = prueba$Diabetes, Predicho = predicci n)
print(mc)

# Calcular la exactitud
exactitud <- sum(diag(mc)) / sum(mc)
cat("Exactitud en conjunto prueba:", round(exactitud * 100, 2), "%\n")
```

Listing 55: Modelo de árbol de decisión - diabetes - datos balanceados.

```
library(rpart)
```

```
library(rpart.plot)
library(dplyr)
dat_salud_idw_mod$ECVMaldef <- as.factor(dat_salud_idw_mod$ECVMaldef)
dat_salud_idw_mod$COD_DISTRITO <- as.factor(dat_salud_idw_mod$COD_
    DISTRITO)
dat_salud_idw_mod$COD_SEXO <- as.factor(dat_salud_idw_mod$COD_SEXO)
dat_salud_idw_mod$DESC_TRAMO_FARMACIA <- as.factor(dat_salud_idw_mod$
    DESC_TRAMO_FARMACIA)
set.seed(19354)
library(caret)
trainIndex <- createDataPartition(dat_salud_idw_mod$ECVMaldef, p = 0.8,
     list = FALSE)
entrenamiento_original <- dat_salud_idw_mod[trainIndex, ]
prueba <- dat_salud_idw_mod[-trainIndex, ]
clase0 <- entrenamiento_original %>% filter(ECVMaldef == "0")
clase1 <- entrenamiento_original %>% filter(ECVMaldef == "1")
n_min <- min(nrow(clase0), nrow(clase1))
set.seed(123)
clase0_sub <- clase0 %>% sample_n(n_min)
clase1_sub <- clase1 %>% sample_n(n_min)
entrenamiento_balanceado <- bind_rows(clase0_sub, clase1_sub)
table(entrenamiento_balanceado$ECVMaldef)
modelo_ECVMaldef <- rpart(ECVMaldef ~ ., data = entrenamiento_
    balanceado, method = "class")
png("Directorio.png", width = 1000, height = 800)
rpart.plot(modelo_ECVMaldef, cex = 2)
title(main = " rbol de Decisi n - ECVMaldef vs Contaminantes (
    Balanceado)", cex.main = 2)
dev.off()
predicci n <- predict(modelo_ECVMaldef, prueba, type = "class")
mc <- table(Real = prueba$ECVMaldef, Predicho = predicci n)
exactitud <- sum(diag(mc)) / sum(mc)
cat("Exactitud en conjunto prueba:", round(exactitud * 100, 2), "%\n")
```

Listing 56: Modelo de árbol de decisión - ECV mal definida - datos balanceados.

```
library(rpart)
library(rpart.plot)
library(dplyr)
dat_salud_idw_mod$Cardiopatiisquemica <- as.factor(dat_salud_idw_mod$
    Cardiopatiisquemica)
dat_salud_idw_mod$COD_DISTRITO <- as.factor(dat_salud_idw_mod$COD_
    DISTRITO)
```

```
dat_salud_idw_mod$COD_SEXO <- as.factor(dat_salud_idw_mod$COD_SEXO)
dat_salud_idw_mod$DESC_TRAMO_FARMACIA <- as.factor(dat_salud_idw_mod$
    DESC_TRAMO_FARMACIA)
set.seed(19354)
library(caret)
trainIndex <- createDataPartition(dat_salud_idw_mod$Cardiopatiisquemica
    , p = 0.8, list = FALSE)
entrenamiento_original <- dat_salud_idw_mod[trainIndex, ]
prueba <- dat_salud_idw_mod[-trainIndex, ]
clase0 <- entrenamiento_original %>% filter(Cardiopatiisquemica == "0")
clase1 <- entrenamiento_original %>% filter(Cardiopatiisquemica == "1")
n_min <- min(nrow(clase0), nrow(clase1))
set.seed(123)
clase0_sub <- clase0 %>% sample_n(n_min)
clase1_sub <- clase1 %>% sample_n(n_min)
entrenamiento_balanceado <- bind_rows(clase0_sub, clase1_sub)
table(entrenamiento_balanceado$Cardiopatiisquemica)
modelo_Cardiopatiisquemica <- rpart(Cardiopatiisquemica ~ ., data =
    entrenamiento_balanceado, method = "class")
png("Directorio.png", width = 1000, height = 800)
rpart.plot(modelo_Cardiopatiisquemica, cex = 2)
title(main = " rbol de Decisi n - Cardiopatiisquemica vs
    Contaminantes (Balanceado)", cex.main = 2)
dev.off()
predicci n <- predict(modelo_Cardiopatiisquemica, prueba, type = "
    class")
mc <- table(Real = prueba$Cardiopatiisquemica, Predicho = predicci n)
exactitud <- sum(diag(mc)) / sum(mc)
cat("Exactitud en conjunto prueba:", round(exactitud * 100, 2), "%\n")
```

Listing 57: Modelo de árbol de decisión - Cardiopatías isquémicas - datos balanceados.

```
library(rpart)
library(rpart.plot)
dat_salud_idw_mod$Hipertension <- as.factor(dat_salud_idw_mod$
    Hipertension)
modelo_Hipertension <- rpart(Hipertension ~ ., data = dat_salud_idw_mod
    , method = "class")
png("Directorio.png", width = 1000, height = 800)
rpart.plot(modelo_Hipertension,
           type = 4,
           extra = 104,
           box.palette = "RdYlGn",
```

```
            shadow.col = "gray",
            nn = TRUE,
            cex = 1.2,
            main = "  rbol  de Decisi n - Hipertensi n vs Contaminantes
                ")
dev.off()
```

Listing 58: Modelo de árbol de decisión - Hipertensión - datos sin balancear.

```
library(rpart)
library(rpart.plot)
library(dplyr)
dat_salud_idw_mod$Insuficienciacardiaca <- as.factor(dat_salud_idw_mod$
    Insuficienciacardiaca)
dat_salud_idw_mod$COD_DISTRITO <- as.factor(dat_salud_idw_mod$COD_
    DISTRITO)
dat_salud_idw_mod$COD_SEXO <- as.factor(dat_salud_idw_mod$COD_SEXO)
dat_salud_idw_mod$DESC_TRAMO_FARMACIA <- as.factor(dat_salud_idw_mod$
    DESC_TRAMO_FARMACIA)
set.seed(19354)
library(caret)
trainIndex <- createDataPartition(dat_salud_idw_mod$
    Insuficienciacardiaca, p = 0.8, list = FALSE)
entrenamiento_original <- dat_salud_idw_mod[trainIndex, ]
prueba <- dat_salud_idw_mod[-trainIndex, ]
clase0 <- entrenamiento_original %>% filter(Insuficienciacardiaca == "0
    ")
clase1 <- entrenamiento_original %>% filter(Insuficienciacardiaca == "1
    ")
n_min <- min(nrow(clase0), nrow(clase1))
set.seed(123)
clase0_sub <- clase0 %>% sample_n(n_min)
clase1_sub <- clase1 %>% sample_n(n_min)
entrenamiento_balanceado <- bind_rows(clase0_sub, clase1_sub)
table(entrenamiento_balanceado$Insuficienciacardiaca)
modelo_Insuficienciacardiaca<- rpart(Insuficienciacardiaca ~ ., data =
    entrenamiento_balanceado, method = "class")
png("Directorio.png", width = 1000, height = 800)
rpart.plot(modelo_Insuficienciacardiaca, cex = 2)
title(main = "  rbol  de Decisi n - Cardiopatiisquemica vs
    Contaminantes (Balanceado)", cex.main = 2)
dev.off()
predicci n <- predict(modelo_Insuficienciacardiaca, prueba, type = "
```

```
    class")
mc <- table(Real = prueba$Insuficienciacardiaca, Predicho = predicci n
    )
print(mc)
exactitud <- sum(diag(mc)) / sum(mc)
cat("Exactitud en conjunto prueba:", round(exactitud * 100, 2), "%\n")
```

Listing 59: Modelo de árbol de decisión - Insuficiencia cardíaca - datos balanceados.

A.4.3. Bosques aleatorios

En estos códigos se han llevado a cabo una serie de modelos de clasificación mediante *Random Forest* para predecir la presencia de diabetes. En estos códigos se ha aplicado el modelo incluyendo las variables de contaminación (C), balanceando los datos (BD), eliminando las variables de contaminación (SC) y dejando únicamente las concentraciones de contaminantes (CC) en la base de datos.

```
library(randomForest)
library(rpart.plot)
library(data.tree)

set.seed(123)

# Asegurar la variable objetivo como factor
dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)

# Dividir datos en entrenamiento (85%) y prueba (15%)
particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]

# Entrenar modelo random forest
modelo_rf <- randomForest(Diabetes ~ ., data = entrenamiento, ntree =
    100, importance = TRUE)
print(modelo_rf)

# Mostrar importancia de variables y gr fico
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")

# Predecir en conjunto de prueba y matriz de confusion
```

```
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Diabetes)
print(mc_rf)

# Calcular exactitud (accuracy)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")

# Guardar gr fico de importancia de variables
png("Directorio.png",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
dev.off()
```

Listing 60: Modelo de bosques aleatorios - diabetes - C

```
library(randomForest)
library(rpart.plot)
library(data.tree)
tiempo <- system.time({
set.seed(123)

#Eliminacion de los contaminantes de la base de datos
dat_sin_contaminantes <- dat_salud_idw_mod[, !(names(dat_salud_idw_mod)
    %in% c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2"))]



dat_sin_contaminantes$Diabetes <- as.factor(dat_sin_contaminantes$
    Diabetes)
particion <- runif(nrow(dat_sin_contaminantes))
entrenamiento <- dat_sin_contaminantes[particion < 0.85, ]
prueba <- dat_sin_contaminantes[particion >= 0.85, ]
modelo_rf <- randomForest(Diabetes ~ ., data = entrenamiento, ntree =
    100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Diabetes)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
```

```
png("Directorio.png",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
dev.off()
```

Listing 61: Modelo de bosques aleatorios - diabetes - SC

```
library(dplyr)
library(randomForest)
library(rpart.plot)
library(data.tree)
tiempo <- system.time({
  set.seed(123)

  # Exclusion de los contaminantes de la base de datos
  dat_sin_contaminantes <- dat_salud_idw_mod[, !(names(dat_salud_idw_
      mod) %in% c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2"))
      ]

  dat_sin_contaminantes$Diabetes <- as.factor(dat_sin_contaminantes$
      Diabetes)
  particion <- runif(nrow(dat_sin_contaminantes))
  entrenamiento <- dat_sin_contaminantes[particion < 0.85, ]
  prueba <- dat_sin_contaminantes[particion >= 0.85, ]

  # Balanceo de datos
  clase0 <- entrenamiento %>% filter(Diabetes == "0")
  clase1 <- entrenamiento %>% filter(Diabetes == "1")
  n_min <- min(nrow(clase0), nrow(clase1))
  set.seed(123)
  clase0_sub <- clase0 %>% sample_n(n_min)
  clase1_sub <- clase1 %>% sample_n(n_min)
  entrenamiento_balanceado_diabetes <- bind_rows(clase0_sub, clase1_sub
      )

  # Entrenar modelo random forest
  modelo_rf <- randomForest(Diabetes ~ ., data = entrenamiento_
      balanceado_diabetes, ntree = 100, importance = TRUE)
  print(modelo_rf)
  print(importance(modelo_rf))
  varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes (
      Balanceado)")
  predi_rf <- predict(modelo_rf, prueba)
```

```
  mc_rf <- table(Predicci n = predi_rf, Real = prueba$Diabetes)
  print(mc_rf)
  exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
  cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
  png("Directorio.png",
      width = 1000, height = 800)
  varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes (
      Balanceado)")
  dev.off()
})
```

Listing 62: Modelo de bosques aleatorios - diabetes - SC+ DB

```
library(dplyr)
library(randomForest)
library(rpart.plot)
library(data.tree)
tiempo <- system.time({
  set.seed(123)
  dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)
  particion <- runif(nrow(dat_salud_idw_mod))
  entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
  prueba <- dat_salud_idw_mod[particion >= 0.85, ]

  # Balanceo de clases
  clase0 <- entrenamiento %>% filter(Diabetes == "0")
  clase1 <- entrenamiento %>% filter(Diabetes == "1")
  n_min <- min(nrow(clase0), nrow(clase1))
  set.seed(123)
  clase0_sub <- clase0 %>% sample_n(n_min)
  clase1_sub <- clase1 %>% sample_n(n_min)
  entrenamiento_balanceado_diabetes <- bind_rows(clase0_sub, clase1_sub
      )

  # Entrenar modelo random forest
  modelo_rf <- randomForest(Diabetes ~ ., data = entrenamiento_
      balanceado_diabetes, ntree = 100, importance = TRUE)
  print(modelo_rf)
  print(importance(modelo_rf))
  varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes (
      Balanceado)")
  predi_rf <- predict(modelo_rf, prueba)
  mc_rf <- table(Predicci n = predi_rf, Real = prueba$Diabetes)
```

```
  print(mc_rf)
  exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
  cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
  variables (opcional)
  png("Directorio.png",
      width = 1000, height = 800)
  varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes (
      Balanceado)")
  dev.off()
})
```

Listing 63: Modelo de bosques aleatorios - diabetes - C + DB

```
library(randomForest)
library(rpart.plot)
library(data.tree)
set.seed(123)

# Datos de la dislipemia
dat_salud_idw_mod$Dislipemia <- as.factor(dat_salud_idw_mod$Dislipemia)

particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Dislipemia ~ ., data = entrenamiento, ntree =
    100, importance = TRUE)
print(modelo_rf)
varImpPlot(modelo_rf, main = "Importancia de Variables - Dislipemia")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Dislipemia)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
png("Directorio.png",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables - Dislipemia")
dev.off()
```

Listing 64: Modelo de bosques aleatorios - Dislipemia - C

```
library(randomForest)
library(rpart.plot)
library(data.tree)
```

```
1790
set.seed(123)
1792 dat_salud_idw_mod$Hipotiroidismo <- as.factor(dat_salud_idw_mod$
    Hipotiroidismo)
particion <- runif(nrow(dat_salud_idw_mod))
1794 entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]
1796 modelo_rf <- randomForest(Hipotiroidismo ~ ., data = entrenamiento,
    ntree = 100, importance = TRUE)
print(modelo_rf)
1798 print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables - Hipotiroidismo
    ")
1800 predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Hipotiroidismo)
1802 print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
1804 cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")

1806 png("Directorio.png",
     width = 1000, height = 800)
1808 varImpPlot(modelo_rf, main = "Importancia de Variables - Hipotiroidismo
    ")
dev.off()
```

Listing 65: Modelo de bosques aleatorios - Hipotiroidismo - C

```
1810 library(randomForest)
library(rpart.plot)
1812 library(data.tree)

1814 set.seed(123)
dat_salud_idw_mod$Trastornoestadoanimo <- as.factor(dat_salud_idw_mod$
    Trastornoestadoanimo)
1816 particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
1818 prueba <- dat_salud_idw_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Trastornoestadoanimo ~ ., data =
    entrenamiento, ntree = 100, importance = TRUE)
1820 print(modelo_rf)
print(importance(modelo_rf))
1822 varImpPlot(modelo_rf, main = "Importancia de Variables -
    Trastornoestadoanimo")
```

```
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$
    Trastornoestadoanimo)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")


png("Directorio.png",
     width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables -
    Trastornoestadoanimo")
dev.off()
```

Listing 66: Modelo de bosques aleatorios - Trastorno de ánimo - C

```
library(randomForest)
library(rpart.plot)
library(data.tree)

set.seed(123)
dat_salud_idw_mod$Trastornodeansiedad <- as.factor(dat_salud_idw_mod$
    Trastornodeansiedad)
particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Trastornodeansiedad ~ ., data = entrenamiento
    , ntree = 100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables -
    Trastornodeansiedad")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$
    Trastornodeansiedad)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")


png("Directorio.png",
     width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables -
```

```
    Trastornodeansiedad")
dev.off()
```

Listing 67: Modelo de bosques aleatorios - trastorno de ansiedad - C

```
1858 library(randomForest)
     library(rpart.plot)
1860 library(data.tree)

1862 set.seed(123)
     dat_salud_idw_mod$Arteriopatideextremidades <- as.factor(dat_salud_idw_
         mod$Arteriopatideextremidades)
1864 particion <- runif(nrow(dat_salud_idw_mod))
     entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
1866 prueba <- dat_salud_idw_mod[particion >= 0.85, ]
     modelo_rf <- randomForest(Arteriopatideextremidades ~ ., data =
         entrenamiento, ntree = 100, importance = TRUE)
1868 print(modelo_rf)
     print(importance(modelo_rf))
1870 varImpPlot(modelo_rf, main = "Importancia de Variables -
         Arteriopatideextremidades")
     predi_rf <- predict(modelo_rf, prueba)
1872 mc_rf <- table(Predicci n = predi_rf, Real = prueba$
         Arteriopatideextremidades)
     print(mc_rf)
1874 exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
     cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
1876


1878 png("Directorio.png",
          width = 1000, height = 800)
1880 varImpPlot(modelo_rf, main = "Importancia de Variables -
         Arteriopatideextremidades")
     dev.off()
```

Listing 68: Modelo de bosques aleatorios - Arteriopatía de extremidades - C

```
1882 library(randomForest)
     library(rpart.plot)
1884 library(data.tree)

1886 set.seed(123)
     dat_salud_idw_mod$Cardiopatiisquemica <- as.factor(dat_salud_idw_mod$
         Cardiopatiisquemica)
```

```
particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Cardiopatiisquemica ~ ., data = entrenamiento
    , ntree = 100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables -
    Cardiopatiisquemica")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$
    Cardiopatiisquemica)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")


png("Directorio.png",
     width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables -
    Cardiopatiisquemica")
dev.off()
```

Listing 69: Modelo de bosques aleatorios - Cardiopatías isquémicas - C

```
library(randomForest)
library(rpart.plot)
library(data.tree)

set.seed(123)
dat_salud_idw_mod$Hipertension <- as.factor(dat_salud_idw_mod$
    Hipertension)
particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Hipertension ~ ., data = entrenamiento, ntree
     = 100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables - Hipertension")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Hipertension)
print(mc_rf)
```

```
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")

png("Directorio.png",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables - Hipertension")
dev.off()
```

Listing 70: Modelo de bosques aleatorios - Hipertensión - C

```
library(randomForest)
library(rpart.plot)
library(data.tree)

set.seed(123)
dat_salud_idw_mod$Insuficienciacardiaca <- as.factor(dat_salud_idw_mod$
    Insuficienciacardiaca)
particion <- runif(nrow(dat_salud_idw_mod))
entrenamiento <- dat_salud_idw_mod[particion < 0.85, ]
prueba <- dat_salud_idw_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Insuficienciacardiaca ~ ., data =
    entrenamiento, ntree = 100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables -
    Insuficienciacardiaca")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$
    Insuficienciacardiaca)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")


png("Directorio.png",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables -
    Insuficienciacardiaca")
dev.off()
```

Listing 71: Modelo de bosques aleatorios - Insuficiencia cardíaca - C

```
library(randomForest)
```

```
library(rpart.plot)
library(data.tree)

set.seed(123)
dat_salud_idw_mod$Dislipemia <- as.factor(dat_salud_idw_mod$Dislipemia)
variables_contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2")
datos_filtrados <- dat_salud_idw_mod[, c("Dislipemia", variables_contaminantes)]
particion <- runif(nrow(datos_filtrados))
entrenamiento <- datos_filtrados[particion < 0.85, ]
prueba <- datos_filtrados[particion >= 0.85, ]
modelo_rf <- randomForest(Dislipemia ~ ., data = entrenamiento, ntree = 100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables - Dislipemia (Contaminantes)")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Dislipemia)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
```

Listing 72: Modelo de bosques aleatorios - Dislipemia - CC

A.4.4. Redes neuronales

En este análisis se entrenaron distintos modelos de red neuronal para predecir la presencia de diabetes, comparando el desempeño según las variables utilizadas. Primero se emplearon todas las variables disponibles, luego solo las relacionadas con contaminación ambiental, y finalmente se excluyeron estas últimas para observar su impacto.

```
library(nnet)
library(caret)
library(ggplot2)
library(lattice)

set.seed(123)
```

```r
# Asegurarse que la variable respuesta es factor
dat_salud_idw_mod$Diabetes <- as.factor(dat_salud_idw_mod$Diabetes)

# Escalar las variables numericas
preproc <- preProcess(dat_salud_idw_mod[, -which(names(dat_salud_idw_mod) == "Diabetes")], method = c("center", "scale"))
dat_scaled <- predict(preproc, dat_salud_idw_mod)
dat_scaled$Diabetes <- dat_salud_idw_mod$Diabetes  # A adir la variable respuesta

# Dividir datos en entrenamiento (85%) y prueba (15%)
particion <- runif(nrow(dat_scaled))
entrenamiento <- dat_scaled[particion < 0.85, ]
prueba <- dat_scaled[particion >= 0.85, ]

# Entrenar red neuronal
modelo_nn <- nnet(Diabetes ~ ., data = entrenamiento, size = 5, maxit = 200, decay = 0.01, trace = FALSE)

# Predecir en conjunto de prueba
predi_nn_prob <- predict(modelo_nn, prueba, type = "raw")
predi_nn <- ifelse(predi_nn_prob > 0.5, "1", "0")
predi_nn <- factor(predi_nn, levels = levels(prueba$Diabetes))

# Matriz de confusion
mc_nn <- confusionMatrix(predi_nn, prueba$Diabetes)
print(mc_nn$table)

# Exactitud
cat("Exactitud en conjunto prueba:", round(mc_nn$overall["Accuracy"] * 100, 2), "%\n")
```

Listing 73: Modelo de redes neuronales - diabetes - C

```r
library(nnet)
library(caret)
library(ggplot2)
library(lattice)

set.seed(123)

# Mantener unicamente los contaminantes
variables_contaminantes <- c("CO", "NO2", "O3_1", "O3_26", "PM10", "
```

```
    PM25", "SO2")
datos_filtrados <- dat_salud_idw_mod[, c("Diabetes", variables_
    contaminantes)]


datos_filtrados$Diabetes <- as.factor(datos_filtrados$Diabetes)
preproc <- preProcess(datos_filtrados[, -which(names(datos_filtrados)
    == "Diabetes")], method = c("center", "scale"))
dat_scaled <- predict(preproc, datos_filtrados)
dat_scaled$Diabetes <- datos_filtrados$Diabetes
particion <- runif(nrow(dat_scaled))
entrenamiento <- dat_scaled[particion < 0.85, ]
prueba <- dat_scaled[particion >= 0.85, ]
modelo_nn <- nnet(Diabetes ~ ., data = entrenamiento, size = 5, maxit =
     200, decay = 0.01, trace = FALSE)
predi_nn_prob <- predict(modelo_nn, prueba, type = "raw")
predi_nn <- ifelse(predi_nn_prob > 0.5, "1", "0")
predi_nn <- factor(predi_nn, levels = levels(prueba$Diabetes))
mc_nn <- confusionMatrix(predi_nn, prueba$Diabetes)
print(mc_nn$table)
cat("Exactitud en conjunto prueba:", round(mc_nn$overall["Accuracy"] *
    100, 2), "%\n")
```

Listing 74: Modelo de redes neuronales- diabetes - CC

```
library(nnet)
library(caret)
library(ggplot2)
library(lattice)

set.seed(123)

# Eliminacion de contaminantes
dat_sin_contaminantes <- dat_salud_idw_mod[, !(names(dat_salud_idw_mod)
     %in% c("CO", "NO2", "O3_1", "O3_26", "PM10", "PM25", "SO2"))]
dat_sin_contaminantes$Diabetes <- as.factor(dat_sin_contaminantes$
    Diabetes)
preproc <- preProcess(dat_sin_contaminantes[, -which(names(dat_sin_
    contaminantes) == "Diabetes")], method = c("center", "scale"))
dat_scaled <- predict(preproc, dat_sin_contaminantes)
dat_scaled$Diabetes <- dat_sin_contaminantes$Diabetes
particion <- runif(nrow(dat_scaled))
entrenamiento <- dat_scaled[particion < 0.85, ]
```

```
prueba <- dat_scaled[particion >= 0.85, ]
modelo_nn <- nnet(Diabetes ~ ., data = entrenamiento, size = 5, maxit =
    200, decay = 0.01, trace = FALSE)
predi_nn_prob <- predict(modelo_nn, prueba, type = "raw")
predi_nn <- ifelse(predi_nn_prob > 0.5, "1", "0")
predi_nn <- factor(predi_nn, levels = levels(prueba$Diabetes))
mc_nn <- confusionMatrix(predi_nn, prueba$Diabetes)
print(mc_nn$table)
cat("Exactitud en conjunto prueba:", round(mc_nn$overall["Accuracy"] *
    100, 2), "%\n")
```

Listing 75: Modelo de bosques aleatorios - diabetes - C

A.4.5. Base de datos IDW^6

Debido a que la base de datos utilizada anteriormente era aquella en la que la base de datos eran interpolados por el método de IDW en los siguientes códigos se aplican bosques aleatorios con la bsae de datos de IDW^6.

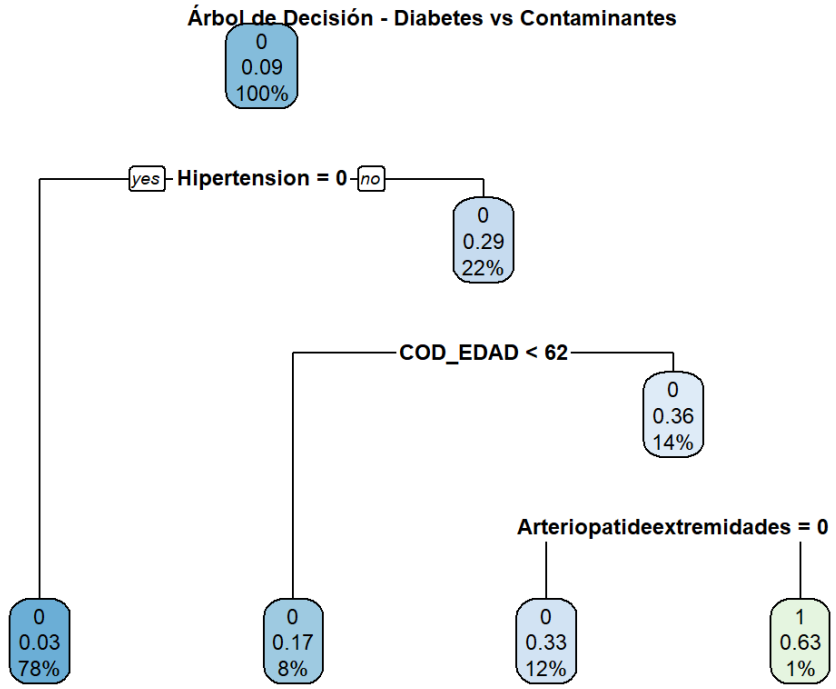
```
library(randomForest)
library(rpart.plot)
library(data.tree)

set.seed(123)

dat_salud_idw6_mod$Diabetes <- as.factor(dat_salud_idw6_mod$Diabetes)
particion <- runif(nrow(dat_salud_idw6_mod))
entrenamiento <- dat_salud_idw6_mod[particion < 0.85, ]
prueba <- dat_salud_idw6_mod[particion >= 0.85, ]
modelo_rf <- randomForest(Diabetes ~ ., data = entrenamiento, ntree =
    100, importance = TRUE)
print(modelo_rf)
print(importance(modelo_rf))
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
predi_rf <- predict(modelo_rf, prueba)
mc_rf <- table(Predicci n = predi_rf, Real = prueba$Diabetes)
print(mc_rf)
exac_rf <- sum(diag(mc_rf)) / sum(mc_rf)
cat("Exactitud en conjunto prueba:", round(exac_rf * 100, 2), "%\n")
png("Directorio.png",
    width = 1000, height = 800)
varImpPlot(modelo_rf, main = "Importancia de Variables - Diabetes")
```

```
dev.off()
```

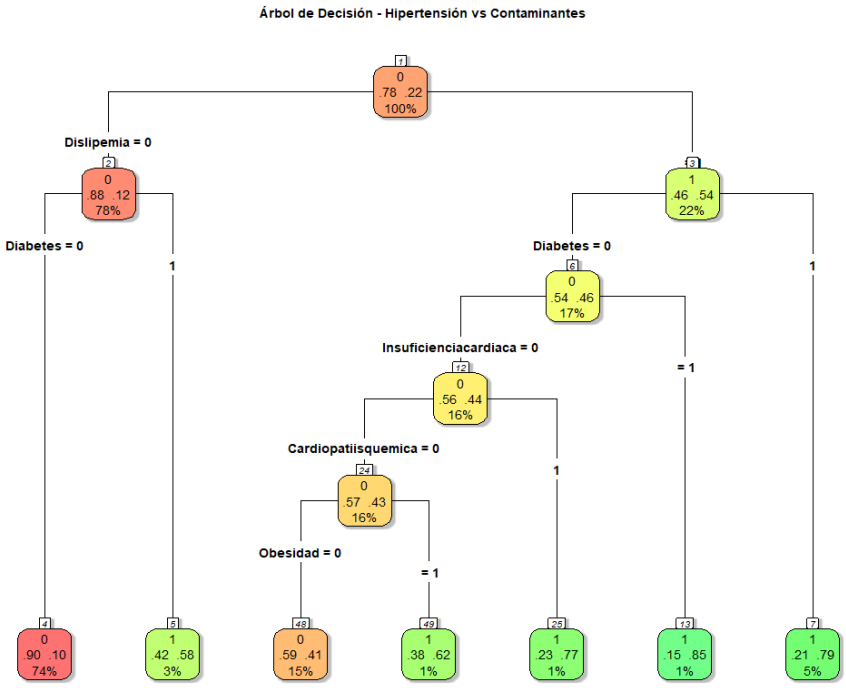
Listing 76: Modelo de bosques aleatorios - diabetes - IDW6

```
library(nnet)
library(caret)
library(ggplot2)
library(lattice)

set.seed(123)


dat_salud_idw6_mod$Diabetes <- as.factor(dat_salud_idw6_mod$Diabetes)
preproc <- preProcess(dat_salud_idw6_mod[, -which(names(dat_salud_idw6_
    mod) == "Diabetes")], method = c("center", "scale"))
dat_scaled <- predict(preproc, dat_salud_idw6_mod)
dat_scaled$Diabetes <- dat_salud_idw6_mod$Diabetes
particion <- runif(nrow(dat_scaled))
entrenamiento <- dat_scaled[particion < 0.85, ]
prueba <- dat_scaled[particion >= 0.85, ]
modelo_nn <- nnet(Diabetes ~ ., data = entrenamiento, size = 5, maxit =
     200, decay = 0.01, trace = FALSE)
predi_nn_prob <- predict(modelo_nn, prueba, type = "raw")
predi_nn <- ifelse(predi_nn_prob > 0.5, "1", "0")
predi_nn <- factor(predi_nn, levels = levels(prueba$Diabetes))
mc_nn <- confusionMatrix(predi_nn, prueba$Diabetes)
print(mc_nn$table)
cat("Exactitud en conjunto prueba:", round(mc_nn$overall["Accuracy"] *
    100, 2), "%\n")
```

Listing 77: Modelo de redes neuronales - diabetes - IDW6



(a) Diabetes



(b) Hipertensión

Figura 61: Resultado de aplicación de árboles de decisión a algunas enfermedades, con resultados. Referencias: elaboración propia

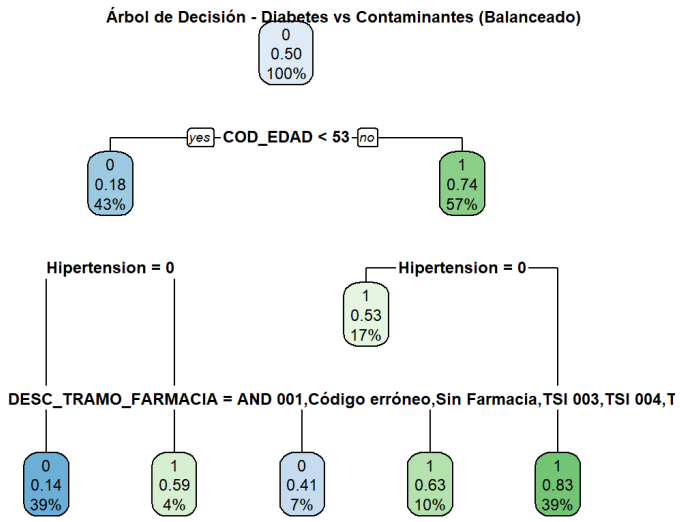
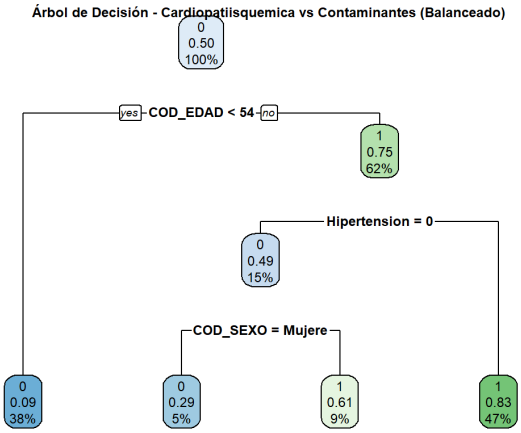
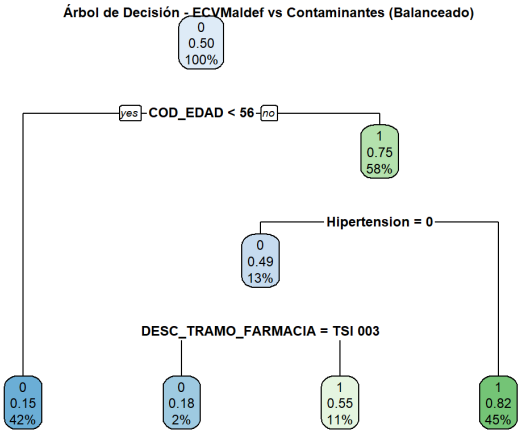


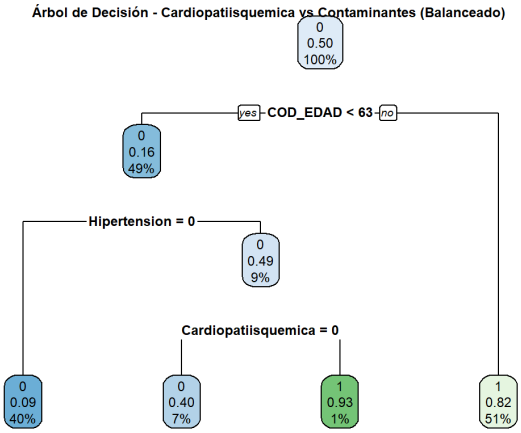
Figura 62: Árbol de decisión para la diabetes con los datos balanceados por submuestreo. Referencias: elaboración propia



(a) Cardiopatías isquémicas



(b) ECV mal definido



(c) Insuficiencia cardíaca

Figura 63: Resultados de árboles de decisión con los datos balanceados. Referencias: elaboración propia

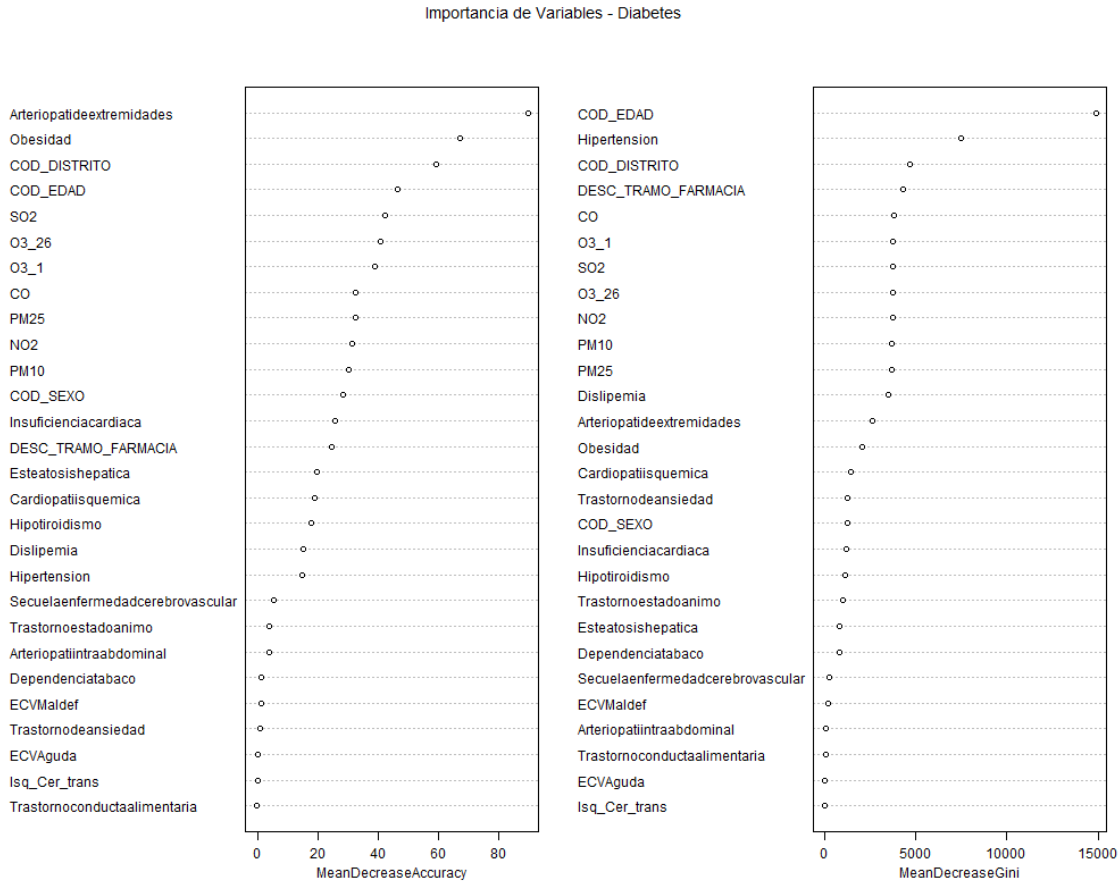


Figura 64: Variables más importantes en aplicación de bosques aleatorios para la diabetes con contaminantes. Referencias: elaboración propia

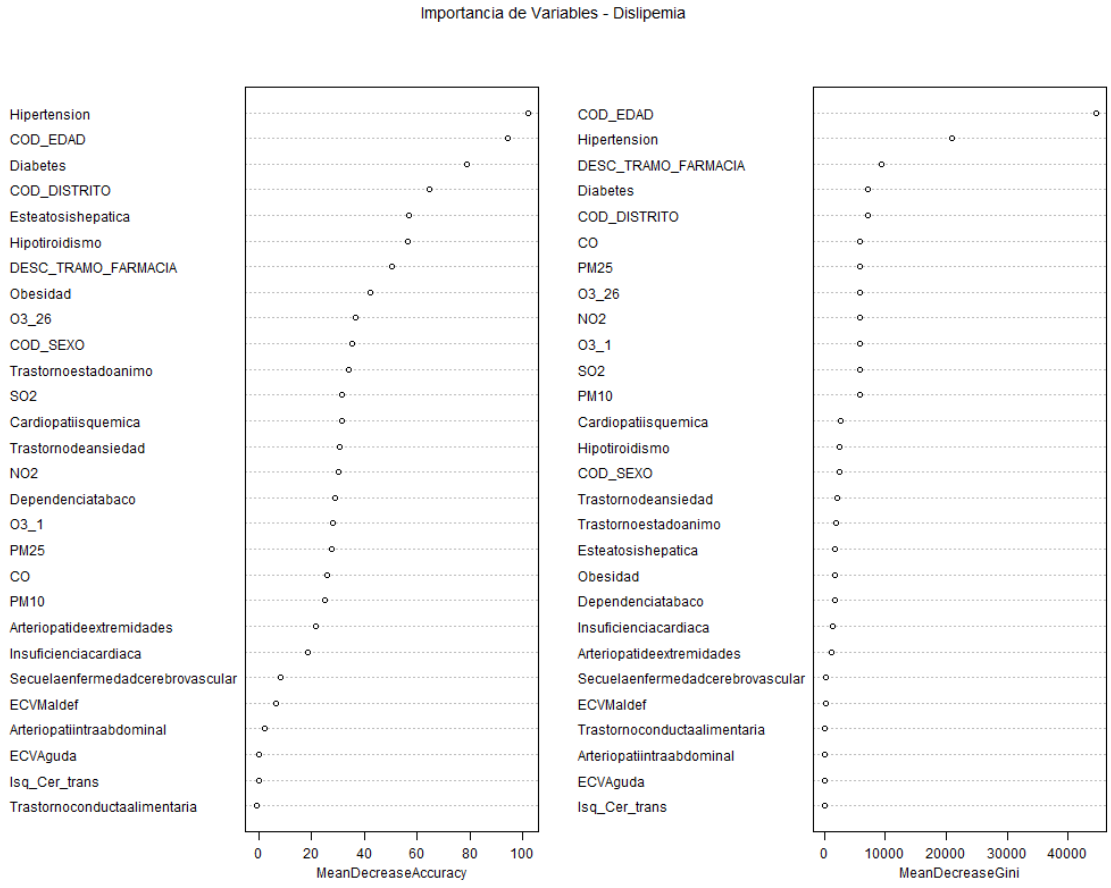


Figura 65: Variables más importantes en aplicación de bosques aleatorios para la dislipemia. Referencias: elaboración propia

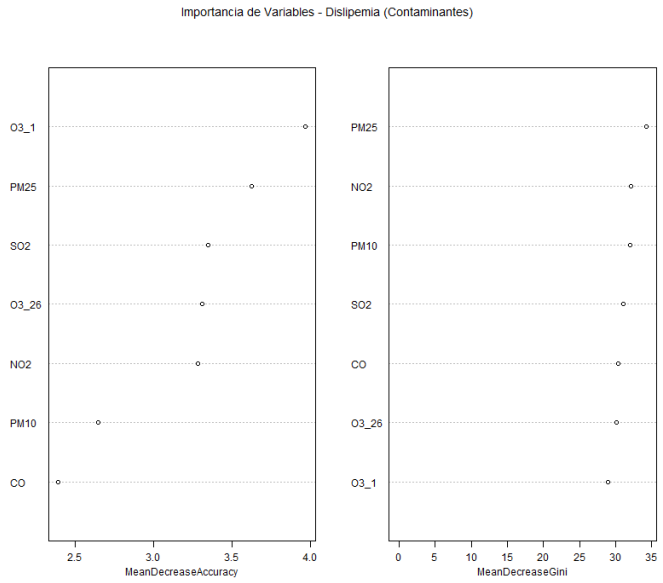


Figura 66: Variables más importantes en aplicación de bosques aleatorios para la dislipemia con la única presencia de contaminantes. Referencias: elaboración propia

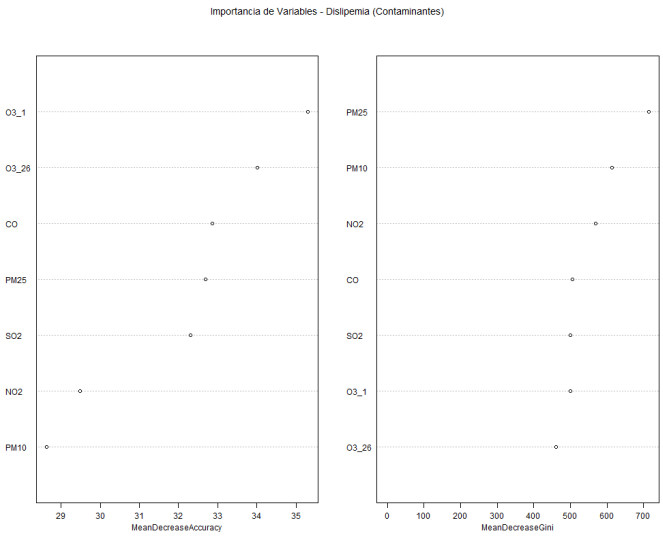


Figura 67: Variables más importantes en aplicación de bosques aleatorios para la dislipemia con la única presencia de contaminantes y datos balanceados. Referencias: elaboración propia

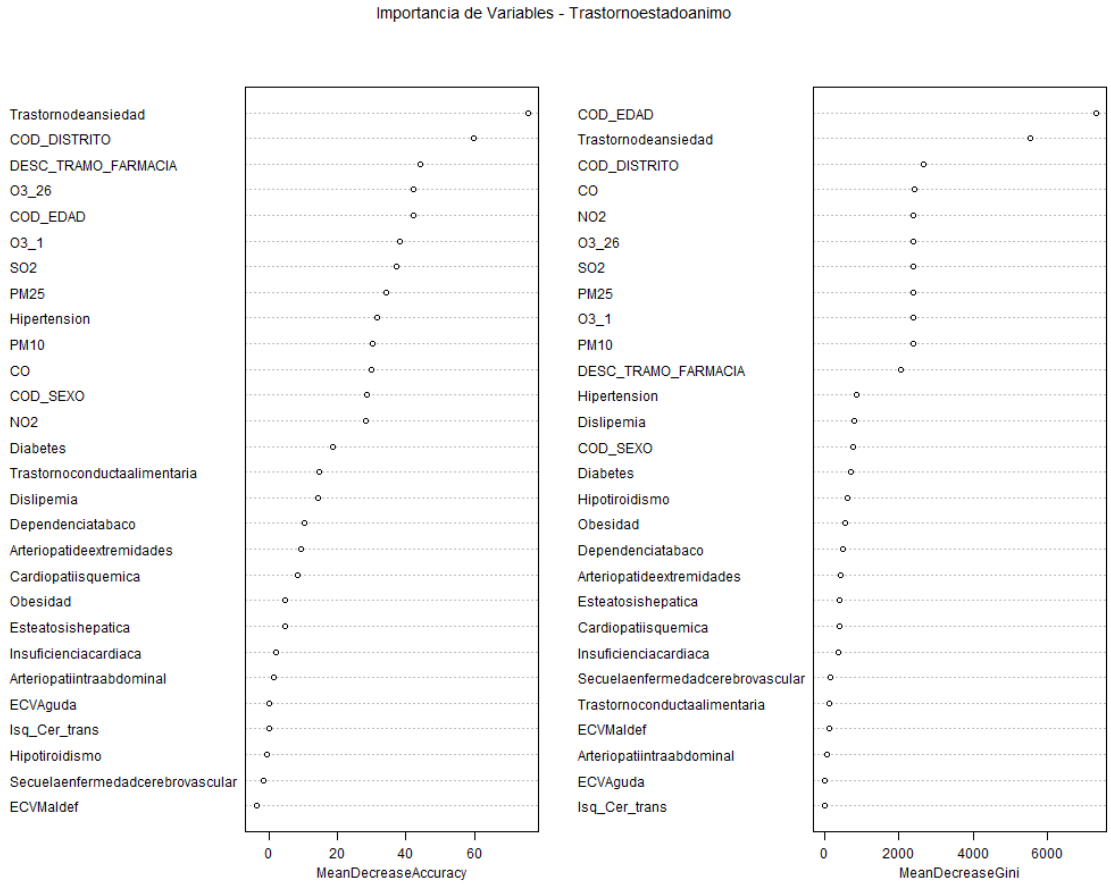


Figura 68: Variables más importantes en aplicación de bosques aleatorios para el trastorno del ánimo. Referencias: elaboración propia

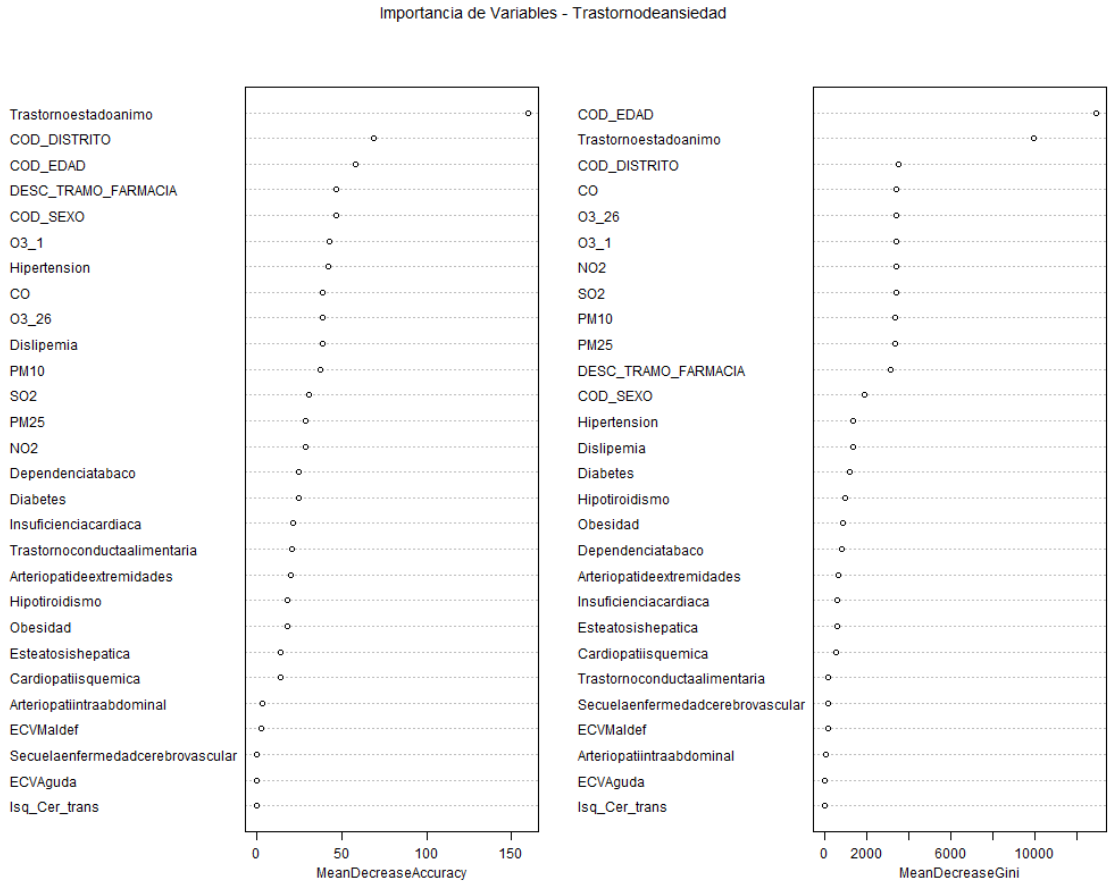


Figura 69: Variables más importantes en aplicación de bosques aleatorios para el trastorno de ansiedad. Referencias: elaboración propia

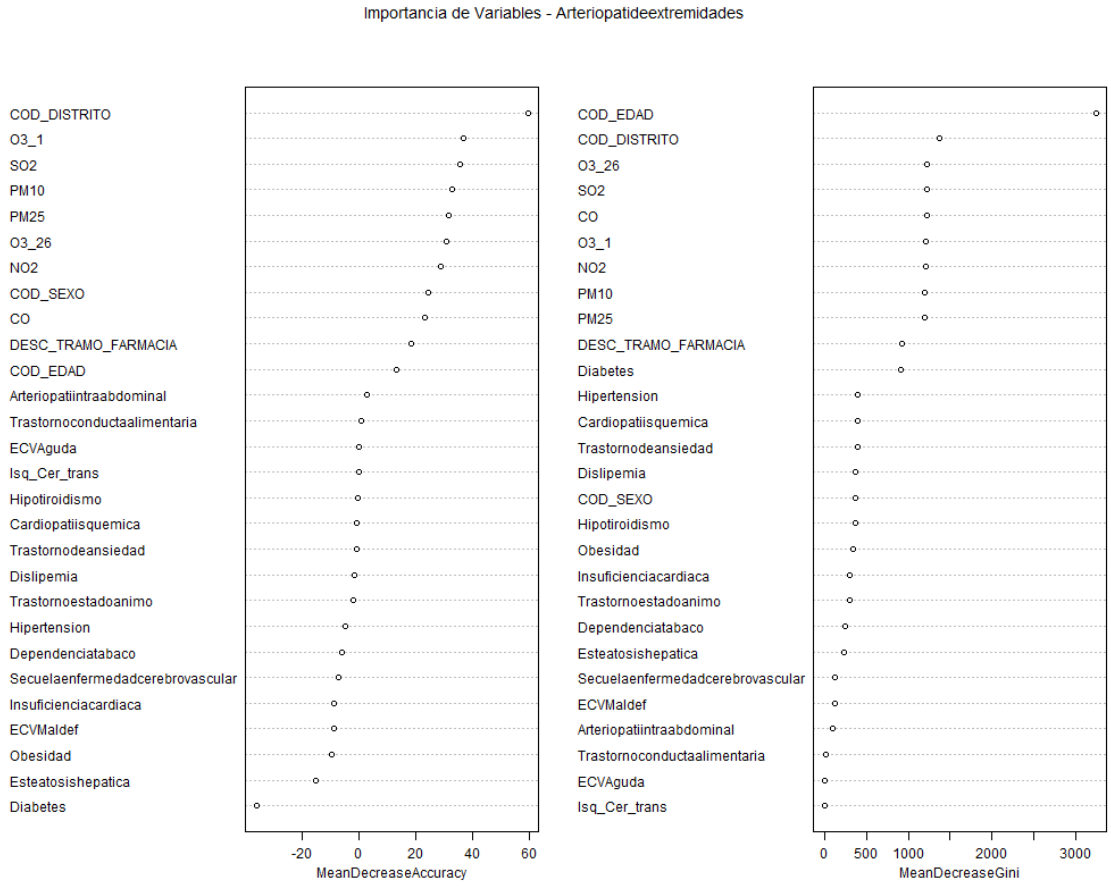


Figura 70: Variables más importantes en aplicación de bosques aleatorios para la arteriopatía de extremidades. Referencias: elaboración propia

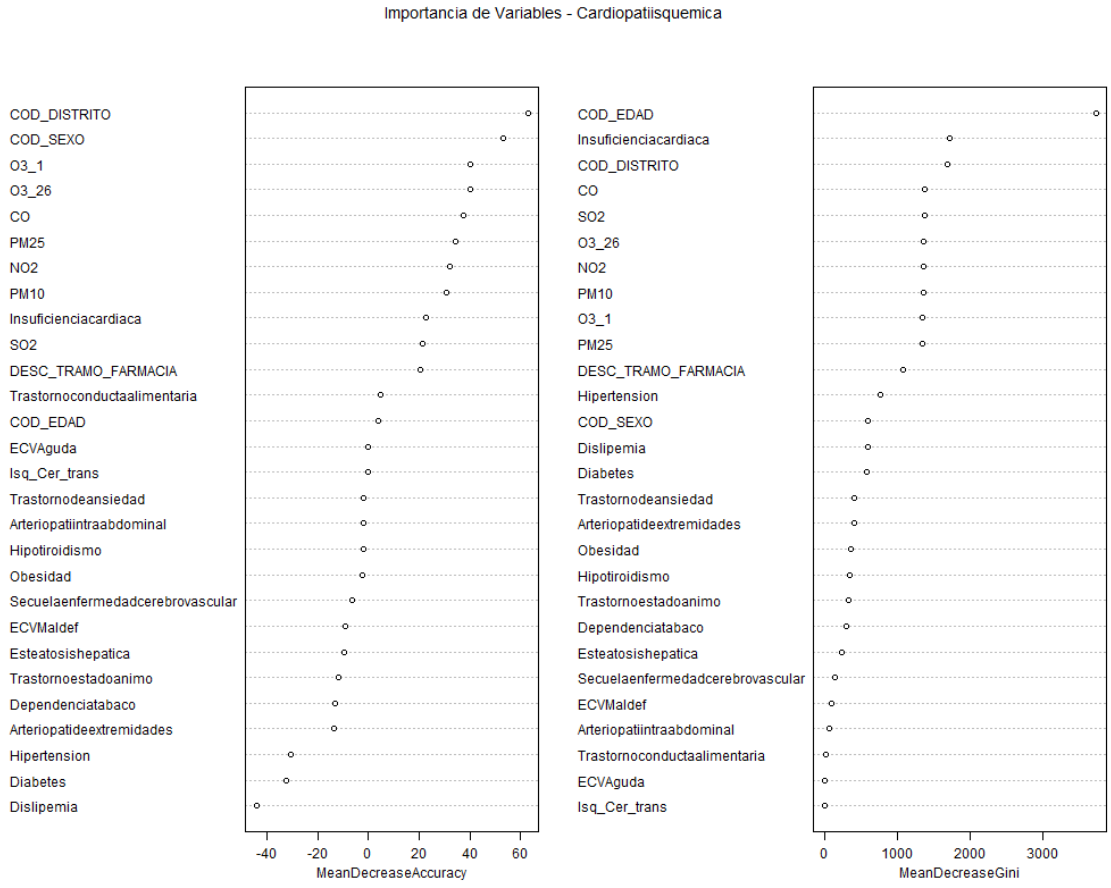


Figura 71: Variables más importantes en aplicación de bosques aleatorios para las cardiopatías isquémicas. Referencias: elaboración propia

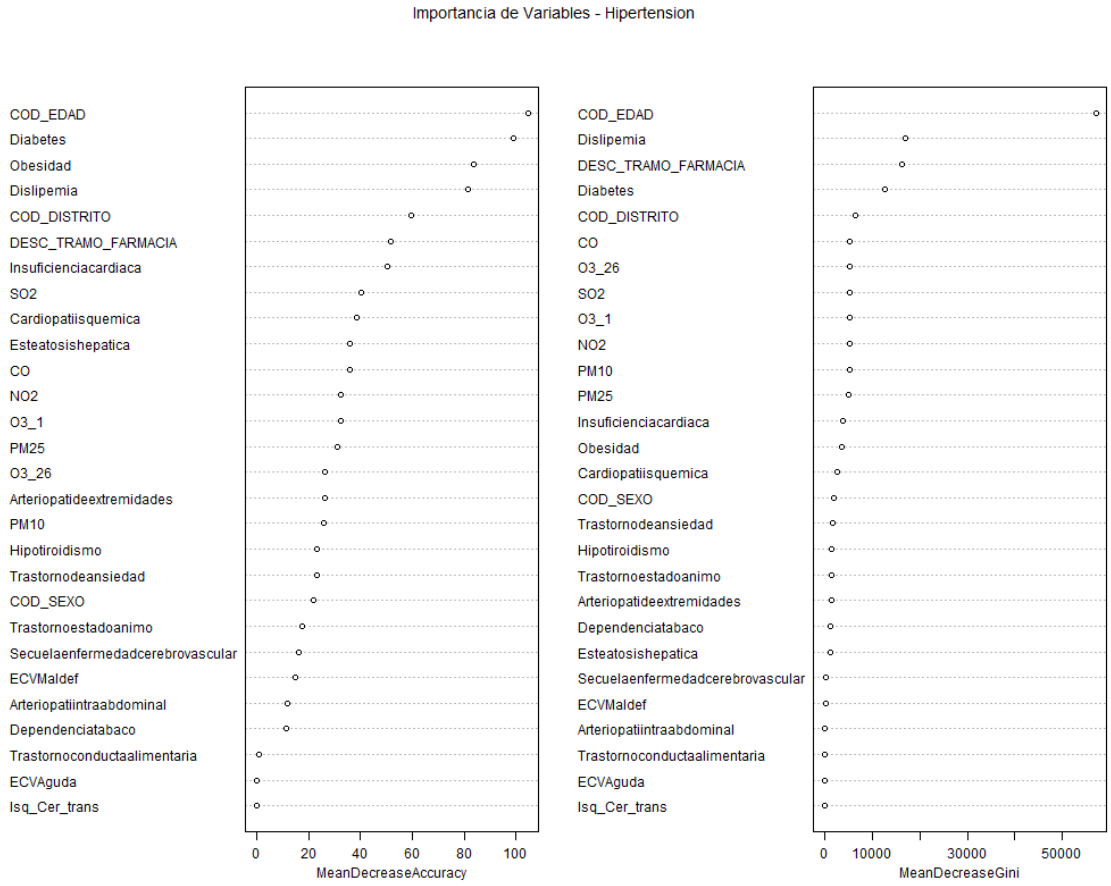


Figura 72: Variables más importantes en aplicación de bosques aleatorios para la hipertensión. Referencias: elaboración propia

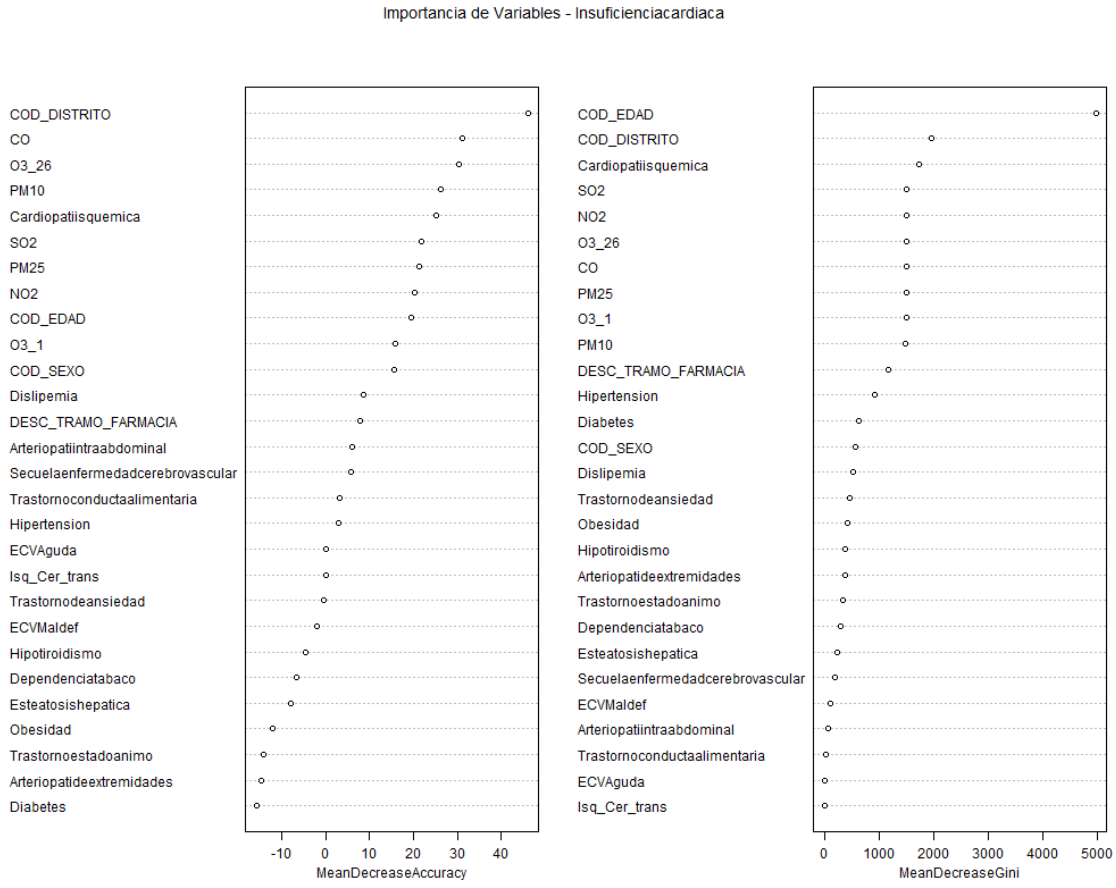


Figura 73: Variables más importantes en aplicación de bosques aleatorios para la insuficiencia cardíaca. Referencias: elaboración propia

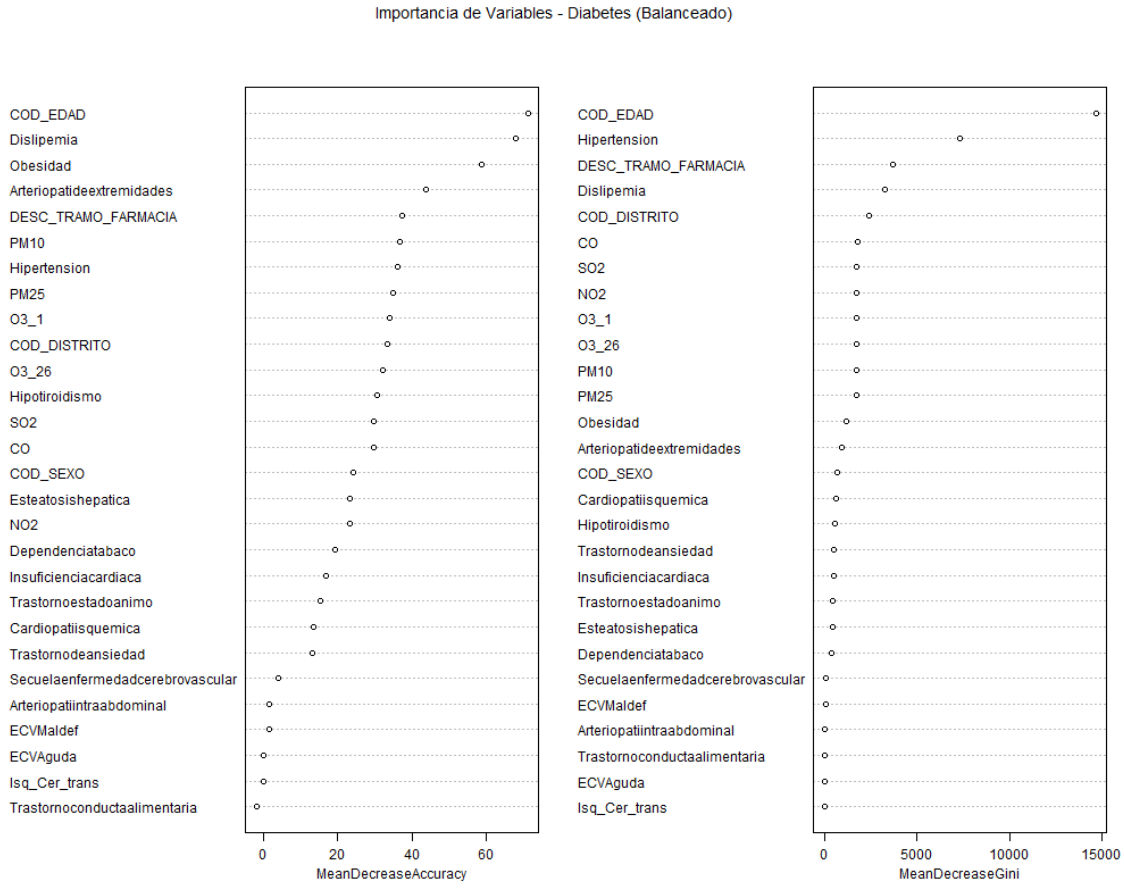


Figura 74: Variables más importantes en aplicación de bosques aleatorios para la diabetes con contaminantes y datos balanceados. Referencias: elaboración propia

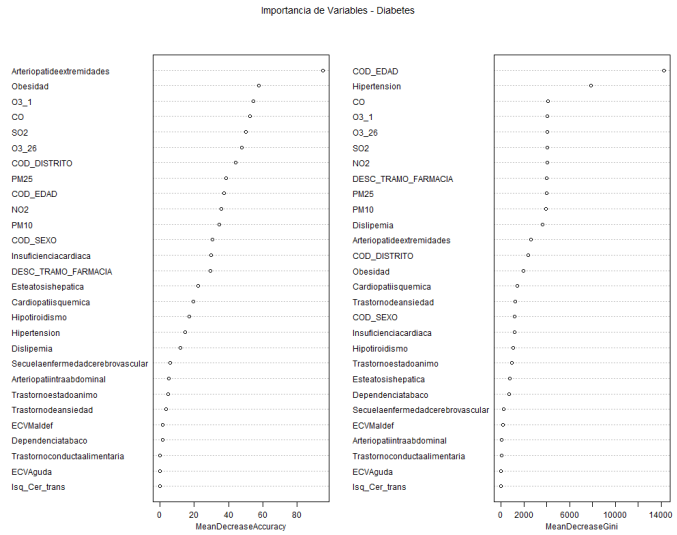


Figura 75: Variables más importantes en aplicación de bosques aleatorios para la diabetes utilizando el interpolador IDW^6. Referencias: elaboración propia