UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS EN TOPOGRAFÍA, GEODESIA Y CARTOGRAFÍA
TITULACIÓN DE GRADO EN INGENIERÍA DE LAS TECNOLOGÍAS DE LA INFORMACIÓN GEOESPACIAL

TRABAJO FIN DE GRADO

Predicción de la calidad de vida centrada en la popularización de las "Ciudades de 15 minutos" y basada en características geoespaciales y algoritmos de aprendizaje automático. Simulación, ensayo o implementación en el municipio de Castellar del Vallès (Cataluña)

Alumno: Alejandro Gómez Sierra

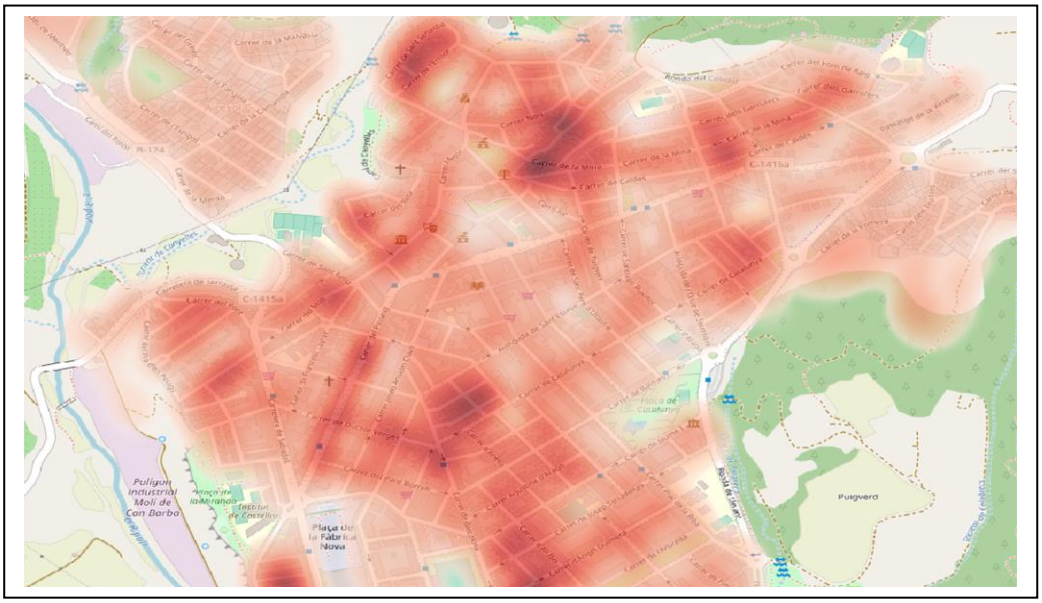
Tutores:
Calimanut-Ionut Cira
Miguel Ángel Manso Callejo

Madrid, Junio de 2023

UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS EN TOPOGRAFÍA, GEODESIA Y CARTOGRAFÍA
TITULACIÓN DE MÁSTER EN INGENIERÍA GEODÉSICA Y CARTOGAFÍA

TRABAJO FIN DE GRADO

Predicción de la calidad de vida centrada en la popularización de las "Ciudades de 15 minutos" y basada en características geoespaciales y algoritmos de aprendizaje automático. Simulación, ensayo o implementación en el municipio de Castellar del Vallès (Cataluña)



Alumno: Alejandro Gómez Sierra

Tutores:
Calimanut-Ionut Cira
Miguel Ángel Manso Callejo

Madrid, Junio, 2023

AGRADECIMIENTOS

En primer lugar, quiero agradecerle este proyecto a mi familia, por apoyarme todos estos años de etapa universitaria y en especial a mi madre, que siempre ha estado a mi lado y quien me animó a estudiar esta Carrera. Sin su apoyo incondicional nada de esto habría sido posible.

Gracias a todos mis profesores durante estos años y especialmente a mis tutores, Ionut y Miguel Ángel, por guiarme en este proyecto y darme las pautas para enfocarlo correctamente y poder mejorarlo.

Gracias a todo el equipo de Berger-Levrault, en especial a Jordi Valeriano y a Jordi Martínez, por confiar en mí para realizar mis primeras prácticas en Berger-Levrault y por darme esta fantástica idea de proyecto, proporcionándome una ayuda inestimable.

Por último, quiero agradecer a mis compañeros y amigos que siempre hayan estado ahí, sirviendo de gran apoyo en muchos momentos tanto dentro como fuera de la Universidad.

ABSTRACT

Urban planning has become increasingly important in recent years, due to the mobility restrictions associated with the global pandemic caused by COVID-19 and the concept of “15 minutes cities” has gained a significant importance. “15 minutes cities” promote that most basic needs and daily services can be reached on foot in fifteen minutes, or less, from any point in the town. In parallel, it can also be observed that an increased number of applications based on artificial intelligence that are capable of solving complex tasks more efficiently than humans, are being lately proposed.

This project studies the quality of life in the municipality of Castellar del Vallès (Barcelona, Catalonia) using artificial intelligence techniques and geospatial characteristics. The objective is the obtention of a machine learning model capable of accurately predicting the quality of life for each dwelling in the studied municipality. The characteristics used for training were obtained from a survey conducted on a sample $n = 41$ persons, with ages between 15 and 70 years. The questionnaire revealed the importance of eleven groups of indicators (for example, rest areas, cultural spaces, educational, sports and health centers, etc.) for the studied sample. The importance calculated was later used to propose a formula for calculating the quality of life (QOL) that is based on the considered indicators.

Afterwards, several popular artificial intelligence algorithms were implemented and optimized to identify the most suitable one for predicting QOL for the studied town. The results obtained by each algorithm will be analyzed based on appropriate performance metrics for regression tasks, the best performing one being XGBoost (it achieved an R2 score of 0.9992). The corresponding model was later implemented to obtain additional graphs and a heat map to achieve a better visualization of the distribution of the QOL within the municipality.

This entire project has been implemented in the Python programming language using open-source libraries.

Keywords: quality of life prediction, 15 minutes cites, machine learning, geospatial indicators

RESUMEN

El urbanismo ha adquirido una gran importancia en los últimos años debido a las restricciones de movilidad asociadas con la pandemia causada por el COVID 19 y el concepto de “ciudades de quince minutos” ha obtenido una importancia significativa. Las ciudades de quince minutos promueven que la mayor parte de las necesidades básicas y los servicios diarios puedan ser accesibles a pie en quince minutos, o menos, desde cualquier punto de la ciudad. Al mismo tiempo, últimamente están surgiendo un número cada vez mayor de aplicaciones basadas en la inteligencia artificial que pueden solucionar tareas complejas de manera más eficiente que los humanos.

Este proyecto estudia la calidad de vida en el municipio de Castellar del Vallès (Barcelona, Cataluña) utilizando técnicas de inteligencia artificial y características geoespaciales. El objetivo es la obtención de un modelo de aprendizaje automático capaz de predecir de manera ajustada la calidad de vida de cada hogar del municipio. Las características utilizadas para la preparación se obtuvieron de una encuesta basada en una muestra $n = 41$ personas, con edades comprendidas entre los quince y los setenta años. El cuestionario reveló la importancia de once grupos de indicadores (por ejemplo, áreas de descanso, espacios culturales, centros educativos, deportivos y de salud, etc.) para dicha muestra. La importancia calculada fue posteriormente utilizada para proponer una fórmula para calcular la calidad de vida que está basada en los indicadores tenidos en cuenta.

Posteriormente, varios algoritmos populares de inteligencia artificial fueron implementados y optimizados para identificar el más adecuado para predecir calidad de vida para el municipio estudiado. Los resultados obtenidos por cada algoritmo serán analizados basados en las métricas de rendimiento apropiadas para tareas de, la mejor de ellas XGBoost, que consiguió una puntuación R2 de 0,9992. El modelo correspondiente fue puesto en práctica posteriormente para obtener gráficos adicionales y un mapa de calor para conseguir una mayor visualización de la distribución de la calidad de vida dentro del municipio. Todo el proyecto ha sido llevado a cabo en el lenguaje de programación Python utilizando librerías de código abierto.

Palabras clave: predicción de calidad de vida, ciudades de 15 minutos, aprendizaje automático, indicadores geoespaciales

ÍNDICE DE CONTENIDOS

LISTA DE TABLAS

LISTA DE FIGURAS

ACRÓNIMOS Y ABREVIATURAS

- **OMS.** Organización Mundial de la Salud
- **ONU.** Organización de las Naciones Unidas
- **SIG.** Sistemas de Información Geográfica
- **QOL.** Calidad De Vida
- **HRQOL.** Calidad De Vida Relacionada con la Salud
- **IDH.** Índice de desarrollo Humano
- **PNUD.** Programa de las Naciones Unidas para el Desarrollo
- **IHDI.** Índice de Desarrollo Humano ajustado por la Desigualdad
- **LEI.** Índice de Esperanza de Vida
- **IE.** Índice de Educación
- **MYSI.** Índice de Años Promedio de Escolaridad
- **EYSI.** Índice de Años Esperados de Escolaridad
- **II.** Índice de renta
- **IMF.** Informe Mundial de la Felicidad
- **UE.** Unión Europea
- **INE.** Instituto Nacional de Estadística
- **IA.** Inteligencia Artificial
- **ANI.** Inteligencia Artificial Estrecha
- **AGI.** Inteligencia Artificial General
- **ASI.** Inteligencia Artificial Superior
- **ML.** Aprendizaje Automático
- **DL.** Aprendizaje Profundo
- **SVM.** Support Vector Machine
- **SVR.** Support Vector Regression
- **KNN.** K-Nearest Neighbors
- **CART.** Árbol de Clasificación y Regresión
- **MDI** Mean Decrease in Impurity
- **MDA** Mean Decrease Accuracy
- **OOB.** Out Of Bag
- **NaN.** Valor nulo
- **SGD.** Descenso de Gradiente Estocástico
- **AoS.** Amount of Say

- ***GBDT.*** *Gradient Boosting Decision Trees*
- ***POI.*** Punto de Interés
- **FOSS.** Software Libre y de Código Abierto
- **BBDD.** Bases de Datos
- ***SO.*** Sistema Operativo
- ***SAAS.*** Software As A Service
- ***TFG.*** Trabajo de Fin de Grado

1. INTRODUCCIÓN

1.1. Antecedentes

La calidad de vida siempre ha sido un tema importante para los gobiernos de distintos países [1], siendo un tema a tratar en varias ocasiones para la Organización Mundial de la Salud (OMS) [2] y para la Organización de las Naciones Unidas (ONU) [3]. Pero la calidad de vida siempre ha sido un tema complicado debido a que esta no es fácil de medir, ya que la percepción que se tiene sobre ella es algo muy subjetivo [4]. Debido a esto, distintas organizaciones han probado a medirla con diferentes métodos intentando así obtener un resultado lo más objetivo posible.

Además de esto, Kent Larson presentó en 2012 su proyecto de ciudades de 20 minutos [5] como un plan para mejorar la planificación del urbanismo de las ciudades y, a partir de ese momento, varios expertos han presentado diferentes propuestas a lo largo de todos estos años, siendo la ciudad de 15 minutos de Carlos Moreno [6] la propuesta más destacada actualmente. Este campo ha empezado a tomar más importancia en estos años más recientes porque, debido a la reciente pandemia mundial provocada por la COVID-19, se han producido unas limitaciones de movilidad que han mostrado el problema de distribución de servicios en muchos núcleos de población [7] [8].

Por otra parte, la inteligencia artificial ha empezado a tomar mucha importancia en estos últimos años, debido a la rápida evolución que se está produciendo en esta y como nos está afectando [9]. El uso y los límites de la inteligencia artificial es actualmente uno de los debates principales que hay en la sociedad, especialmente con la salida de ChatGPT, llegando al punto de publicarse una carta firmada por más de 1000 empresas pidiendo parar el desarrollo de inteligencia artificial por ser esta una amenaza para la humanidad [9]. Este es un tema que puede dar lugar a debate pero lo que es innegable es que ya hay aplicaciones que funcionan con inteligencia artificial que son capaces de realizar trabajos igual o mejor que los propios humanos, utilizando esta herramienta en varios campos, como para realizar predicciones a corto plazo o resolver problemas actuales [10].

1.2. Motivación

Durante todos estos años de carrera he visto muchos campos de los que abarca esta, pero he sentido especial interés por la Inteligencia Artificial y los Sistemas de Información Geográfica (SIG). Sabiendo esto, y teniendo en cuenta los antecedentes propuestos en el apartado 1.1, la idea de realizar un proyecto capaz de abarcar estos dos campos es muy llamativa.

Además, aún no se ha utilizado la inteligencia artificial para estudiar la calidad de vida por lo que el proyecto que se plantea para resolver este problema se vuelve innovador ya que explora una nueva forma de abordar este problema.

1.3. Objetivos

El objetivo principal de este proyecto es responder a las preguntas de si una inteligencia artificial es capaz de realizar un análisis geoespacial de un municipio para calcular la calidad de vida asociada a ese análisis, y si este valor que calcula es tan bueno como el de un humano y cuánto tiempo tarda en calcularlo, para ver si utilizar esta herramienta es más eficiente que calcularlo por un método tradicional. Para responder estas preguntas el proyecto se dividirá en varias etapas:

- **Aplicar técnicas de análisis geoespacial:** En el primer punto, se realizará un análisis de distancias entre puntos de interés y accesos a viviendas para poder ver qué puntos hay a menos de 15 minutos de cada vivienda. Una vez calculadas las distancias, se analizarán los puntos que hay en este rango de tiempo en cada vivienda para obtener un valor distinto de la calidad de vida asociada a estas.
- **Realización de un cuestionario para obtener una fórmula utilizable para el cálculo de la calidad de vida.**
- **Entrenamiento e Implementación de algoritmos de inteligencia artificial:** Una vez calculada el valor de la calidad de vida para los puntos del análisis espacial, el conjunto de datos etiquetados con el valor QOF se utilizará para el entrenamiento y la evaluación de los algoritmos de inteligencia artificial. En esta etapa se entrenarán varios algoritmos para comprobar cuál es el que ofrece mejores resultados, si los resultados que este ofrece son válidos para predecir en otro lugar, y si el tiempo que tarda en calcularlos es suficientemente corto como para preferir este método frente al cálculo tradicional.

2. Marco teórico

2.1. Calidad de vida

La OMS define la calidad de vida (QOL) como “la percepción que tiene un individuo de su posición en la vida en el contexto de la cultura y los sistemas de valores en los que vive y en relación con sus metas, expectativas, estándares y preocupaciones” [2]. Para poder medir la QOL, se utilizan varios indicadores para poder aproximarse mejor y estos varían según quién haga el estudio, aun así, hay varios indicadores que están más estandarizados y aparecen en la mayoría de los estudios, estos son: riqueza, empleo, medio ambiente, salud física y mental, educación, recreación y tiempo libre, pertenencia social, creencias religiosas, seguridad, seguridad y libertad [11].

2.1.1. Introducción

Además de esto, la QOL aparece en muchos contextos diferentes, por ejemplo, uno de los más comunes es la calidad de vida relacionada con la salud (HRQOL) [12], este concepto mide la QOL basándose mayoritariamente en factores que afectan a la salud. También hay que destacar un artículo de la revista *Applied Research in the Quality of Life* que habla de la teoría comprometida [13], esta es una forma diferente de medir la QOL. Esta teoría establece cuatro dominios: ecología, economía, política y cultura; y a su vez cada dominio tiene sus indicadores. A partir de esta teoría han nacido otras variantes que recogen otros dominios aparte de los ya mencionados, como la libertad, la felicidad y la seguridad humana.

La QOL, a diferencia de otros términos, como la densidad de población o el PIB per cápita, no se puede medir de una forma completamente objetiva, ya que como se ha visto antes, algunos indicadores como la felicidad son muy subjetivos. Por lo que, para hacer una evaluación más objetiva, los investigadores dividen la QOL en bienestar emocional y evaluación de la vida [4]. El bienestar emocional se centra en la calidad de sus experiencias emocionales cotidianas: la frecuencia e intensidad de sus experiencias. La evaluación de la vida se basa en evaluar la vida en general en base a una escala. Este es uno de los métodos más comunes utilizados actualmente para medir la QOL, pero además de este hay muchos más métodos diferentes, como la relación que establece entre la QOL y la productividad el banco de la reserva federal de Kansas

[14] u otros métodos que tratan de medirla en otros términos, como de atención médica o riqueza. Pero estos métodos tratan de medir la parte que permite una medición más objetiva, pero la expresión de los deseos es mucho más difícil. Una forma de medirlo es ver el alcance los ideales de los individuos. Esto es porque no tienen los mismos ideales los ciudadanos de países desarrollados que los de países en desarrollo ya que, un ciudadano de un país en desarrollo aprecia más las necesidades básicas de salud y educación que un ciudadano de un país desarrollado [15].

El economista Robert Constanza escribió en 2008 un artículo llamado "Un enfoque integrador para la medición, investigación y políticas de calidad de vida" [1] en el que dice: *"Si bien la calidad de vida (QOL) ha sido durante mucho tiempo un objetivo político explícito o implícito, la definición y medición adecuadas han sido difíciles de alcanzar. Diversos indicadores "objetivos" y "subjetivos" en una variedad de disciplinas y escalas, y trabajos recientes sobre encuestas de bienestar subjetivo (SWB) y la psicología de la felicidad han despertado un interés renovado."*

2.1.2. Métodos de medición

Sabiendo esto a continuación, se describirán algunos de los métodos más comunes para medir la calidad de vida:

2.1.2.1. Índice de desarrollo humano

El Índice de Desarrollo Humano (IDH) [3] es un índice estadístico compuesto por la esperanza de vida, educación e indicadores de ingreso per cápita. Este indicador utiliza estos indicadores para clasificar a los países en cuatro niveles del desarrollo humano. El IDH es el utilizado por el Programa de las Naciones Unidas para el Desarrollo (PNUD) para calcular el desarrollo de los países en su Informe sobre Desarrollo Humano [3], esto hace que sea la medida de desarrollo internacional más utilizada. La Ilustración 1 muestra un mapa del mundo coloreando los países según la puntuación que han obtenido del IDH, el color varía en incrementos de 0,05 y va desde el color verde oscuro siendo los valores más altos, de 0,95 o superior, hasta el color rojo oscuro siendo los valores más bajos, de 0,399 o inferior, además de colorear en gris los países cuyos datos no estén disponibles. Los datos en los que se basa son de 2021 publicados en 2022.

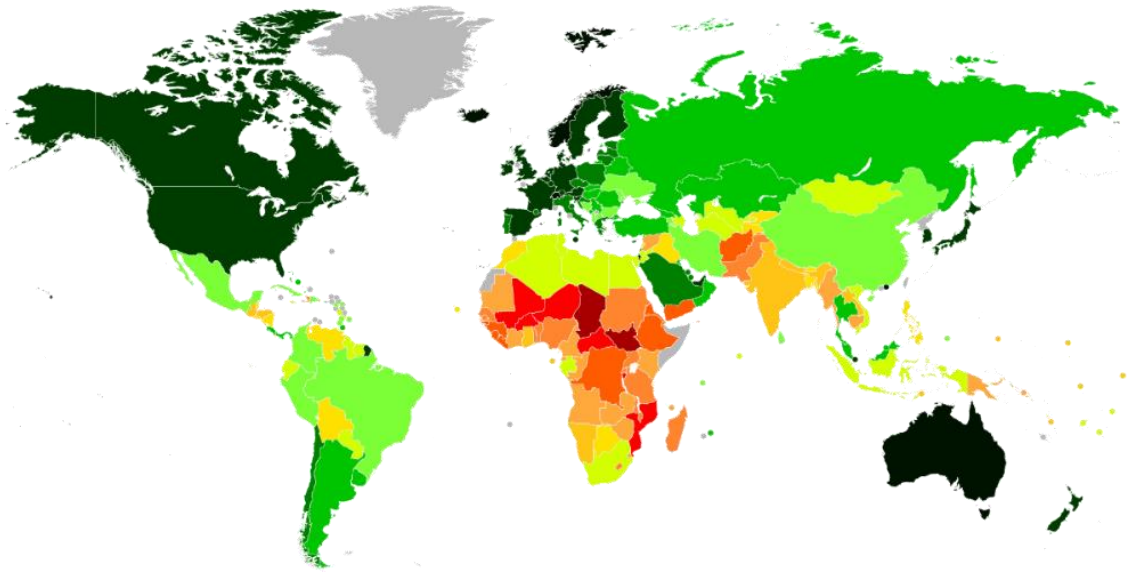


Ilustración 1. Mapa mundial de países por puntajes del IDH. Fuente: [16]

El IDH fue desarrollado en 1990 por el economista Mahbub ul Haq [17] para medir el desarrollo de un país por la Oficina del Informe sobre Desarrollo Humano del PNUD. Para desarrollar este índice dio otro enfoque al trabajo de Amartya Sen sobre las capacidades humanas.

Sin embargo, en 2010 se introdujo el Índice de Desarrollo Humano ajustado por la Desigualdad (IHDI) [18] convirtiendo a este en el índice que da un nivel más real del desarrollo humano y el IDH pasó a utilizarse como un índice de desarrollo humano potencial que se podría conseguir si no hubiese desigualdad. También hay que destacar que el IDH no tiene en cuenta factores como la riqueza neta per cápita o la calidad relativa de los bienes en un país por lo que algunos de los países más desarrollados, como los miembros del G7, tienen posiciones más bajas en el ranking.

Para calcular el IHDI [19] se combinan tres dimensiones: Una vida larga y saludable, que se calcula con la esperanza de vida al nacer, educación que se calcula con el promedio de años de escolaridad y años esperados de escolaridad y un nivel de vida digno que se calcula con el INB per cápita. Para calcular estos tres índices se aplican las ecuaciones (1), (2) y (5):

- **Índice de esperanza de vida (LEI)**

$$LEI = \frac{LE - 20}{85 - 20} = \frac{LE - 20}{65} \quad (1)$$

Siendo LE la esperanza de vida y 85 y 20 son la esperanza de vida máxima y mínima.

- **Índice de educación (IE)**

$$IE = \frac{MYSI + EYSI}{2} \quad (2)$$

Donde MYSI y EYSI se calculan mediante las ecuaciones (3) y (4):

- **Índice de Años Promedio de Escolaridad (MYSI)**

$$MYSI = \frac{MYS}{15} \quad (3)$$

Donde MYS son los años promedio de escolaridad y 15 es el máximo proyectado por este indicador para 2025.

- **Índice de Años Esperados de Escolaridad (EYSI)**

$$EYSI = \frac{EYS}{18} \quad (4)$$

Donde EYS son los años esperados de escolaridad y 18 es el equivalente a lograr una maestría en la mayoría de los países.

- **Índice de renta (II)**

$$II = \frac{\ln(GNIpc) - \ln(100)}{\ln(75000) - \ln(100)} = \frac{\ln(GNIpc) - \ln(100)}{\ln(750)} \quad (5)$$

Donde GNIpc es la renta per cápita y 75000 y 100 son las rentas máxima y mínima.

De esta forma, tras haber calculado el LEI, IE e II, se puede calcular el IDH mediante la fórmula descrita por la ecuación (6):

$$IDH = \sqrt[3]{LEI \cdot IE \cdot II} \quad (6)$$

2.1.2.2. Informe Mundial de la Felicidad

El Informe Mundial de la Felicidad (IMF) [20] es una clasificación de 156 países por niveles de felicidad. El IMF está basado en una encuesta sobre la felicidad nacional cuyos resultados se correlacionan con otros factores de calidad de vida. La felicidad empezó a ser de interés cuando la Asamblea General de la ONU propuso medir la felicidad de los ciudadanos de los países miembros para adoptar mejores políticas públicas en la resolución “65/309 Felicidad: hacia una definición holística del desarrollo” [21] aprobada en julio de 2011. La Ilustración 2 muestra un mapa del mundo coloreado con los distintos niveles de felicidad de 2023.

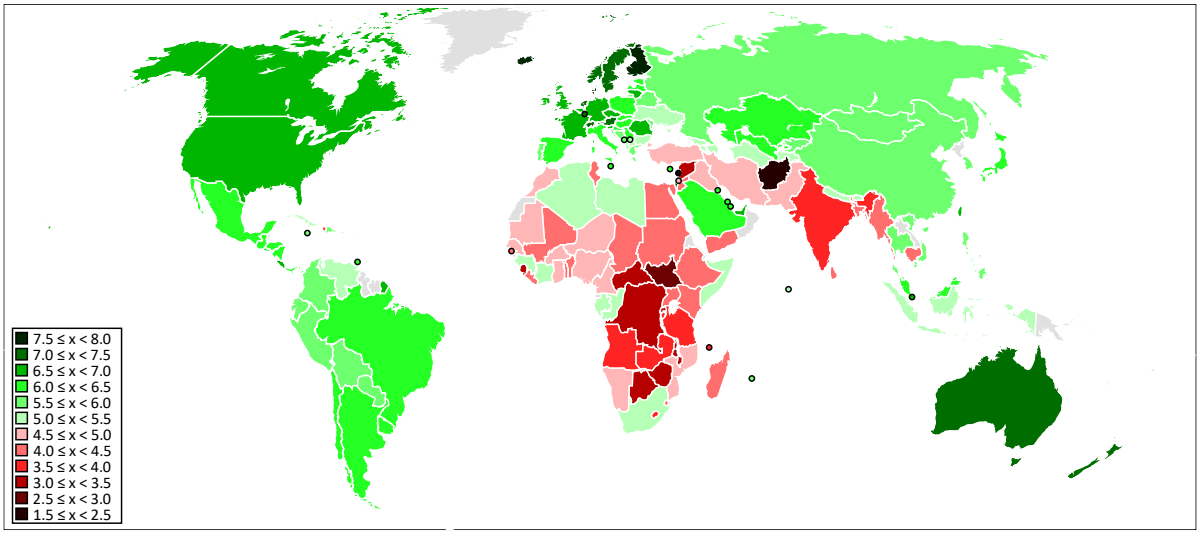


Ilustración 2. Niveles mundiales de felicidad según el Informe Mundial de la Felicidad de 2023. Fuente: [22]

Así, el 1 de abril de 2012 se publicó el primer IMF para ser utilizado en la reunión de Alto Nivel de la ONU presidida por el Secretario General de la ONU Ban Ki-moon y el Primer Ministro Jigmi Thinley de Bután que se llamó "Bienestar y Felicidad: Definiendo un Nuevo Paradigma Económico" [23]. Este informe trató de explicar el estado de la felicidad mundial, mostrando las causas y las implicaciones políticas. El segundo IMF se emitió en 2013 y el tercero en 2015 y, a partir de 2016, se emite un IMF cada 20 de marzo, coincidiendo así con el Día Internacional de la Felicidad de la ONU[24].

La clasificación que realiza el IMF se basa en la encuesta de la escala de Cantrill realizada por la empresa Gallup, Inc [25]. En esta encuesta se pide a los participantes que piensen en una escala que vaya de 0, siendo la peor vida posible, a 10, siendo la mejor vida posible; posteriormente, correlaciona los resultados obtenidos con varios factores para poder explicar las diferencias en las evaluaciones de vida en los distintos países [26]. El uso de medidas subjetivas es un enfoque diferente, ya que permite ver cuál es la perspectiva que se tiene en cada país. En el informe participan expertos de muchos campos para describir como las medidas de bienestar se pueden utilizar para evaluar el progreso de las naciones.

Sin embargo, desde el 2021 el IMF aboga por utilizar los años de vida ajustados al bienestar (WELLBYs) antes que los años de vida ajustados a la calidad (QALYs). Esto es porque los QUALYs solo tienen en cuenta la calidad de vida individual de cada individuo mientras que los WELLBYs tienen en cuenta también la percepción de la vida del individuo, siguiendo así su lema viviendo mucho y viviendo bien [27].

2.1.2.3. Unión Europea

En la Unión Europea (UE) también se ha intentado medir la calidad de vida de los países de una forma objetiva [28]. La UE, ya que tiene acceso a las cuentas nacionales, planteó en un principio utilizar el PIB como medida, pero tras varias observaciones, se llegó a la conclusión de que este no indica nada por sí solo, ya que por ejemplo países como España e Italia tienen un PIB inferior a la media europea, pero son los dos países con mayor esperanza de vida de esta, teniendo estos una esperanza de vida de 83,5 y 83,4 años respectivamente.

Esto ha generado varios debates que han llevado a varias iniciativas, como el informe de J. Stiglitz, A. Sen y J.P. Fitoussi de 2009 sobre la medición del desempeño económico y el progreso social [29] o la Comunicación de la Comisión Europea de 2009 llamada "El PIB y más allá" [30]. El resultado ha sido un consenso en que se necesitan más indicadores que puedan complementar el PIB del país.

El Sistema Estadístico Europeo creó el Grupo de patrocinio sobre la medición del progreso, el bienestar y el desarrollo sostenible. Este grupo trató de desarrollar indicadores que fuesen capaces de responder a los problemas presentados en el informe "El PIB y más allá" [30], presentando su informe en 2011.

En este informe se destaca la necesidad de medir la calidad de vida con un enfoque multidimensional, utilizando indicadores que reflejen la sostenibilidad e indicadores complementarios al PIB. Tras la publicación de este informe, se creó un grupo de expertos, que incluía expertos procedentes de 10 oficinas nacionales de estadística, expertos científicos y representantes de organizaciones internacionales como la OCDE y la Fundación Europea para la Mejora de las Condiciones de Vida y de Trabajo (Eurofound), coordinados por Eurostat cuyo objetivo era definir una serie de indicadores de calidad de vida.

Este grupo se estuvo reuniendo semestralmente desde 2012 hasta 2016 y entregó en 2017 el informe final del grupo de expertos sobre indicadores de calidad de vida [31]. Este informe indica que existen 8+1 dimensiones para medir el bienestar y que estas deben considerarse simultáneamente debido a que haya compensaciones entre ellas. Estas dimensiones son: condiciones de vida materiales, actividad principal, salud, educación, ocio e interacciones sociales, seguridad económica y física, gobernanza y derechos básicos, entorno natural y de vida y experiencia general de la vida.

Las condiciones de vida materiales, la actividad principal y la experiencia general de la vida, a su vez, se dividen en tres subindicadores, el primero se divide en ingresos,

consumo y condiciones materiales; el segundo se divide en cantidad de empleo, calidad del empleo y otra actividad principal, esto incluye la población inactiva y el trabajo no remunerado; y el tercero se divide en satisfacción con la vida, es decir, la apreciación cognitiva que tienes de esta, afecto, estos son los sentimientos o estados emocionales de una persona, tanto positivos como negativos, típicamente medidos con referencia a un punto particular en el tiempo, y eudaimónicos, esto es un sentido de tener significado y propósito en la vida de uno, o buen funcionamiento psicológico.

Esta última dimensión, experiencia general de la vida, corresponde la dimensión que se cuenta aparte (8+1), ya que esta tiene una medición mucho más subjetiva frente al resto de dimensiones.

2.1.2.4. Instituto Nacional de Estadística.

El Instituto Nacional de Estadística (INE) se basa en el informe producido por la UE, por lo que utiliza también 9 indicadores para medir la calidad de vida dentro de un municipio [32]. Estos son: Condiciones materiales de vida, trabajo, salud, educación, ocio y relaciones sociales, seguridad física y personal, gobernanza y derechos básicos, entorno y medioambiente, experiencia general de la vida.

2.2. Ciudades de 15 minutos

La ciudad de 15 minutos [33] es un concepto de planificación urbana que promueve un tipo de ciudad en el que la mayoría de las necesidades básicas y servicios diarios, como el trabajo, las compras y la educación, se pueden alcanzar en 15 minutos o menos a pie o en bicicleta desde cualquier punto de la ciudad. Esta idea también tiene como objetivo reducir el uso del automóvil y con esto promover un desarrollo más sostenible y una mejora del bienestar y la calidad de vida de los habitantes de estas ciudades [34].

2.2.1. Introducción.

Este concepto nace como derivado de otras ideas centradas en la proximidad y accesibilidad para peatones, como la unidad de vecindario de Clarence Perry [35] o el modelo presentado por Jane Jacobs en su artículo llamado La Muerte y Vida de las Grandes Ciudades Americanas [36].

La ciudad de 15 minutos también se ha popularizado más estos años debido a la actual crisis climática y a la pandemia provocada por el virus COVID-19. El Grupo de Liderazgo Climático de Ciudades C40 publica en julio de 2020 un artículo llamado como reconstruir mejor las ciudades de 15 minutos [8]. Algunos de sus puntos son:

- **Actuar con más prioridad según la zona.** *Deben tener más prioridad los barrios con menos ingresos y más desatendidos en los programas de ciudades de 15 minutos. Además, los residentes y comercios locales deben participar en medidas de mejora.*
- **Realizar un proceso de participación inclusivo.** Para poder realizar la actuación definida, algunas ciudades como París, donde el 10% del gasto de la ciudad se determina mediante procesos de presupuestos participativos a nivel de barrio cuyos habitantes tienen la oportunidad de participar en el diseño y la selección de los proyectos que se ejecutarán en este, o Nueva York, donde se han asignado 120 millones de dólares en los últimos ocho años a más de 700 proyectos locales, proponen una serie de presupuestos participativos.
- **Mejorar la infraestructura para caminar y andar en bicicleta.** Estos medios de transporte se han popularizado con la crisis climática y la pandemia provocada por la COVID-19, por lo que es interesante invertir en calles que faciliten esta forma de caminar, ya que así se puede reducir el tráfico y sus emisiones, lo que mejoraría considerablemente la calidad del aire, además de que al reducir el espacio de los automóviles aumenta el espacio para comercios lo que ayuda a recuperar la economía local.
- **Descentralizar servicios básicos para crear barrios más completos**. Este es el principal objetivo de las ciudades de 15 minutos. Aquí se planea garantizar que cada barrio disponga de todos los servicios necesarios para un habitante, esto también contribuirá a reducir la aglomeración en las zonas comerciales. Para cumplir este objetivo se deben garantizar tiendas de alimentos frescos en cada barrio, actualizar los planes que los servicios públicos esenciales, y los espacios verdes sean accesibles a todos los residentes, y promover la vivienda asequible en cada barrio.
- **Aplicar medidas de planificación para facilitar la prosperidad de los barrios.** Además de fomentar comercios y servicios que sean capaces de satisfacer las necesidades básicas a menos de 15 minutos, también se planifica para crear locales donde la población quiera pasar su tiempo. Esto se puede hacer aprovechando las plantas bajas de los edificios, por ejemplo en París la agencia Semaest puede comprar una planta baja de un edificio para reconvertirla en un local comercial, o fomentando el uso flexible de edificios u otros espacios, esto se puede hacer fomentando edificios diseñados para ser convertidos fácilmente para diferentes usos y reduciendo así la necesidad de demolición y reconstrucción, tal y como se está haciendo en París aprovechando los patios

de los colegios o Nueva York permitiendo comercios en los aparcamientos y patios los fines de semana.

- **Fomentar el teletrabajo y la digitalización de los servicios.** Este tipo de medidas ayudarán a limitar la necesidad de desplazamientos, algo que será muy útil en caso de un nuevo brote o la aparición de otro virus. Algunas medidas que pueden adoptar los ayuntamientos para apoyar estas medidas son aumentar los servicios digitales que ofrecen, consultar a directivos de empresas para conocer los obstáculos que tienen estas a la hora de ofrecer teletrabajo, facilitar la adquisición de internet de alta velocidad y promover espacios de trabajo comunitario como las bibliotecas.

En abril de 2020, Massimo Paolini, teórico de arquitectura y miembro de POLLEN, publica en Barcelona el Manifiesto por la reorganización de la ciudad tras la COVID-19 [37], donde habla de cómo reorganizar la ciudad para hacer frente a futuras pandemias. Este manifiesto tiene más de 2000 firmantes tras su última actualización el 17 de julio de 2022. Las medidas que propone se pueden dividir en cuatro objetivos: reorganizar la movilidad, reduciendo el uso del automóvil privado y promoviendo más el uso de la bicicleta y del transporte público, así como la peatonalización progresiva de la ciudad y favoreciendo el juego infantil en las calles y las plazas; (re)naturalizar la ciudad, es decir, incrementar la superficie destinada al verde urbano, esto se puede hacer plantando árboles, reduciendo drásticamente el asfalto sustituyéndolo por materiales porosos que permitan la filtración del agua en el terreno, promoviendo la creación de más espacios verdes y huertos urbano, reduciendo la contaminación lumínica, impulsando la biodiversidad y proviniendo de fuentes de agua potable de calidad en toda la ciudad; des mercantilizar la vivienda a través de varias medidas como ofrecer un hogar individual a personas sin hogar basándose en el modelo finlandés *Housing First*, reducir los precios de alquiler; reducir drásticamente el número de pisos turísticos; impulsar la vivienda pública y garantizar que cada persona mayor pueda seguir viviendo en su hogar; e impulsar el decrecimiento urbano, es decir, una gran reducción del consumo por el impulso de la economía social y local y de los pequeños comercios, esto se puede conseguir eliminando los cruceros, rechazando la construcción de nuevos museos y eliminando cualquier inversión para promocionar la marca Barcelona.

2.2.2. Modelos de ciudad.

La ciudad de 15 minutos es un modelo de ciudad policéntrica, a lo largo del tiempo ha habido varios modelos que proponen este tipo de ciudad y la ciudad de 15 minutos, es una variación de estos modelos.

2.2.2.1. La ciudad de 20 minutos de Kent Larson.

Kent Larson es un arquitecto y actualmente el director del grupo de investigación City Science del MIT Media Lab. En 2012, Kent Larson participó en una charla TED en la que describió el concepto de ciudad de 20 minutos [5]. Aquí explica que su grupo de investigación ha desarrollado una plataforma para simular vecindarios y poder comprobar más fácilmente como actuaría sobre la ciudad distintas implementaciones de diseño, tecnología y política. También, Kent Larson habló de estas ciudades en una clase magistral llamada "Sobre las ciudades" para la Fundación Norman Foster [38], aquí Kent Larson propuso que el planeta se está convirtiendo en una red de ciudades, y que para que estas sean exitosas, se deben convertir en una red de comunidades emprendedoras y de alto rendimiento.

2.2.2.2. La ciudad del minuto T de Luca D'Acci.*

Luca D'Acci introduce en 2013 un nuevo concepto denominado ciudad de T* minutos [39], donde T* es un tiempo no muy largo para llegar a un destino caminando. Este es un enfoque distinto de planificación de urbanismo llamado urbanismo de isobeneficio que está basado en un código morfogenético y propone que haya terrenos naturales, lugares de trabajo, comercios y servicios a menos de un kilómetro o una milla de distancia. Esta ciudad se basa en el código de *Isobenefit Urbanism Morphogenesis* y este es un código que es capaz de simular un escenario de crecimiento urbano permitiendo modificar los valores de sus parámetros relacionados con densidad de población, superficie del terreno, tamaño de la población y factores aleatorios. Este modelo ofrece una gran cantidad de resultados que permiten el urbanismo de isobeneficio [40].

2.2.2.3. El vecindario transitable de 15 minutos de Weng.

En 2019, Weng propone en un artículo el vecindario transitable de 15 minutos [41] con un enfoque en la salud y en las enfermedades no transmisibles, utilizando Shanghái como caso de estudio. Weng sugiere este vecindario como una forma para mejorar la salud de sus habitantes ya que, tal y como argumenta Weng en este caso, las zonas rurales son menos transitables que las zonas urbanas y las zonas con menor transitabilidad tienden a tener un mayor porcentaje de niños. Este modelo se diferencia del resto en que su objetivo es mejorar la salud de los habitantes.

2.2.2.4. La ciudad de 20 minutos de Da Silva

Da silva cita en 2019 la ciudad de Temple en Arizona como un caso de estudio de un espacio urbano donde todas tus necesidades se pueden satisfacer a menos de 20

minutos caminando, bicicleta o transporte público [42]. Da silva se centra en la ciudad de Temple ya que observó que esta ciudad es especialmente accesible en bicicleta, pero esta varía según la ubicación en la que te centres. Este modelo se diferencia del resto en que su objetivo es una mayor accesibilidad dentro del espacio urbano construido.

2.2.2.5. La ciudad de 15 minutos de Carlos Moreno

En 2021, el experto en Urbanismo y profesor de la Universidad Pantheon-Sorbonne de París, Carlos Moreno publica su artículo *Introducing the "15-Minute City"* [6] [7] donde introduce el concepto de ciudades de 15 minutos, este es una forma de organizar la ciudad de tal forma que sus residentes puedan cumplir a menos de 15 minutos seis funciones básicas, las cuales son: vivienda, salud, comercio, salud, educación y entretenimiento. La propuesta de Ciudad de 15 minutos que propone Carlos Moreno se basa en cuatro conceptos fundamentales:

- Proximidad: Los lugares deben de ser cercanos.
- Diversidad: El uso del suelo debe ser variado para proveer una variedad de servicios.
- Densidad: Será necesaria una cantidad de personas para apoyar la diversidad de negocios en un área.
- Ubicuidad: Estos barrios deben ser tan comunes que cualquiera pueda vivir en uno.

Esta propuesta se hizo más popular después de que la alcaldesa de París, Anne Hidalgo hiciese de esta idea el centro de su campaña de reelección a la alcaldía. Para poder conseguir el objetivo de ciudades de 15 minutos, se deben cumplir los siguientes requisitos:

- Las comunidades deben participar en el desarrollo y la implementación de programas municipales.
- Se debe definir y cuantificar quién tiene acceso a qué y dónde lo tiene.
- Se debe involucrar a muchas organizaciones y grupos de personas, como el gobierno o inversores.
- Es importante que este tipo de ciudades sean rentables, por lo que es importante el tipo de comercios que tengamos en ellas.

2.2.3. Implementaciones

Varias ideas que se han visto aquí se han implementado en países de los cinco continentes. Por ejemplo, en Lagos y en Nigeria, se convirtieron escuelas que estaban cerradas en ese momento debido a la pandemia provocada por la COVID-19 en mercados de alimentos para reducir los tiempos de viaje y reforzar los suministros de alimentos dentro de cada comunidad; o en Singapur se han puesto los pueblos de 20 minutos y las ciudades de 45 minutos como objetivo para 2040; en Oregón, se desarrolló en 2012 un plan de viviendas muy similar al modelo de vecindario transitable de 15 minutos de Weng [41] cuyo objetivo es moverse caminando o en bicicleta para mejorar la salud de sus habitantes; o en La ciudad de Melbourne en Australia, que en su plan Melbourne 2017-2050 [43] incluye varios puntos de las ciudades de 15 minutos y plantea la construcción de barrios de 20 minutos.

2.3. Distancias.

A lo largo de este proyecto se verán distintos tipos de distancias, por lo que se definirán todas a continuación para posteriormente, ser solo mencionadas.

2.3.1. Distancia Euclídea

La distancia euclídea [44] es un número positivo que indica la distancia que hay en línea recta entre dos puntos dentro de un mismo espacio euclídeo. Esta distancia se puede deducir a partir del teorema de Pitágoras.

La ecuación (7) muestra cómo es el cálculo de la distancia euclídea entre dos puntos A y B:

$$d_e(A,B) = \sqrt{(X_B - X_A)^2 + (Y_B - Y_A)^2 + (Z_B - Z_A)^2} \quad (7)$$

En la Ilustración 3 se puede ver este mismo cálculo de una forma gráfica:

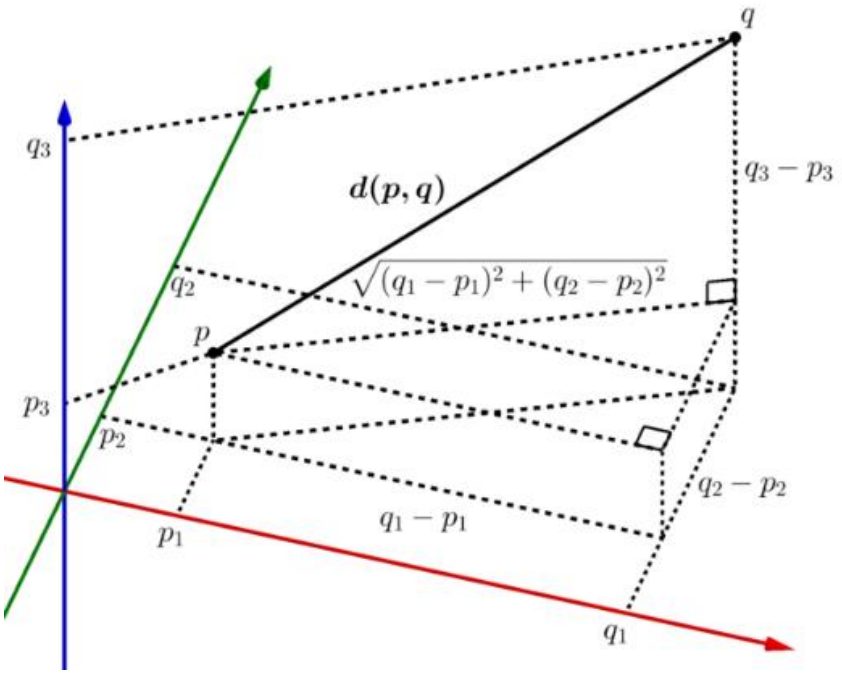


Ilustración 3. Distancia euclídea entre dos puntos. Fuente: [45]

2.3.2. Distancia Manhattan

La distancia Manhattan [46] también es conocida como la geometría del taxista. Creada por Hermann Minkowski en el siglo XIX, esta es la suma de las diferencias absolutas de las coordenadas sobre un plano.

La ecuación (8) muestra cómo es el cálculo de la distancia Manhattan entre dos puntos A y B:

$$d_{Manh}(A,B) = |X_A - X_B| + |Y_A - Y_B| + |Z_A - Z_B| \quad (8)$$

En la Ilustración 4 se puede ver un ejemplo de cómo funciona la distancia Manhattan, siguiendo la ruta a través de los distintos edificios:

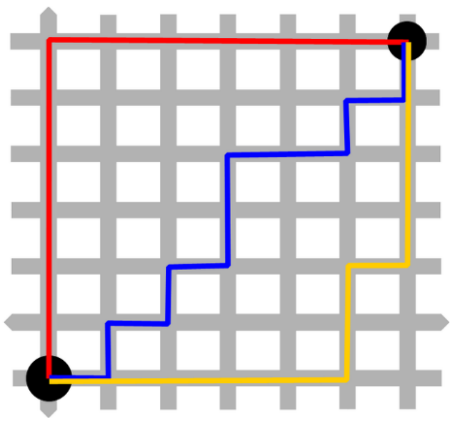


Ilustración 4. Ejemplo de distancia Manhattan entre dos puntos. Fuente: [47]

2.3.3. Distancia Minkowski

La distancia Minkowski [48], que lleva el nombre del matemático Hermann Minkowski, se puede considerar como una generalización de la distancia euclídea y la Manhattan.

La ecuación (10) muestra cómo se calcula la distancia Minkowski de orden p entre los dos puntos descritos en la ecuación (9):

$$X = (x_1, x_2, \dots, x_n)\ y\ Y = (y_1, y_2, \dots, y_n) \in R^n \tag{9}$$

$$d_{Mink}(x, y) = \sum_{i=1}^{n} |x_i - y_i|^p{}^{\frac{1}{p}} \tag{10}$$

Así, la distancia Minkowski es comúnmente usada con valores de p=1 para que coincida con la distancia Manhattan y p=2 para que coincida con la distancia Euclídea.

En la Ilustración 5 se puede ver mediante una serie de gráficas una comparativa de cómo varía la distancia Minkowski para los distintos valores de p:

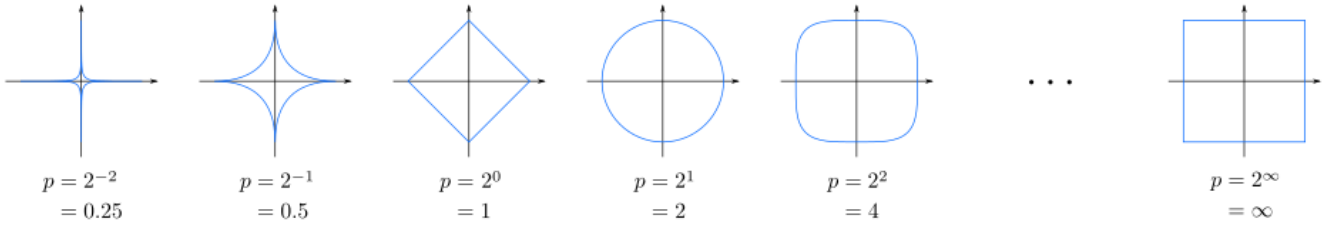


Ilustración 5. Comparativa de cómo funciona la distancia Minkowski para distintos valores de p.
Fuente: [49]

2.4. Inteligencia Artificial

En 2004, John McCarthy definió el término Inteligencia Artificial (IA) en un artículo como "la ciencia y la ingeniería de crear máquinas inteligentes, especialmente programas informáticos inteligentes. Está relacionada con la tarea similar de utilizar ordenadores para comprender la inteligencia humana, pero la IA no se limita a métodos que sean observables biológicamente" [50].

La primera vez que se habla de Inteligencia Artificial (IA) es en la obra publicada en 1950 "Computer Machinery and Intelligence" de Alan Turing [51], quien es conocido como el padre de la informática, comienza el artículo formulando la pregunta "¿pueden pensar las máquinas?". A partir de aquí propone la prueba te Turing, la cual es una

variación de un juego llamado *The Imitation Game.* En este juego se colocan tres personas en tres salas distintas, un hombre, una mujer y un juez, el objetivo es que una de las dos personas pase por el género contrario. A continuación, el juez hace una serie de preguntas para intentar adivinar el género de la persona. El test de Turing actúa de la misma forma, pero cambiando al hombre y la mujer por una persona y una máquina, siendo ahora el objetivo de la máquina pasar por un ser humano y el del juez decidir quién es el humano y quien es la máquina. A partir de aquí, Alan Turing habla de Inteligencia Artificial si la máquina es capaz de pasar la prueba.

Unos años después, en 1995, Stuart Russel y Peter Norvig publicaron su libro *Artificial Intelligence: A Modern Approach* [52]. Este libro se convirtió en uno de los principales manuales para estudiar IA. En esta obra se habla de cuatro posibles objetivos para la IA, que plantea dos enfoques para la forma de actuar de los sistemas:

- Si se ve desde un enfoque humano:
    - Sistemas que piensan como las personas.
    - Sistemas que actúan como las personas.
- Si se ve desde un enfoque ideal:
    - Sistemas que piensan racionalmente.
    - Sistemas que actúan racionalmente.

La IA se puede clasificar de dos formas distintas [53]:

- IA débil: La IA débil también se puede llamar IA estrecha o en inglés *Artificial Narrow Intelligence* (ANI). Esta es una IA entrenada para realizar tareas específicas. Este tipo de IA es la que participa en la mayoría de los casos en los que se utiliza la IA hoy en día. Algunas aplicaciones sólidas de este tipo de IA son los asistentes de voz de Apple, Google o Amazon o los vehículos autónomos.
- IA fuerte: La IA fuerte es una composición de inteligencia artificial general, o en inglés *Artificial General Intelligence* (AGI), e inteligencia artificial superior, o en inglés *Artificial Super Intelligence* (ASI). La AGI es una forma teórica de IA en la que una máquina tiene una inteligencia equivalente a la humana, teniendo así conciencia propia y capacidad de resolver problemas, aprender y planear el futuro. La ASI, también conocida como superinteligencia, es otra forma teórica en la que la IA supera al cerebro humano. De momento la IA fuerte es completamente teórica, aunque se está investigando mucho su desarrollo, por lo que por ahora no es posible encontrar ejemplos de ASI.

Así, se puede decir que la IA utiliza conjuntos de datos para enseñar a una máquina como resolver problemas. Dentro de IA aparecen dos subcategorías denominadas Aprendizaje Automático o *Machine Learning (ML)* y Aprendizaje Profundo o *Deep Learning (DL)*, siendo esta última una subcategoría también del ML [54]. Para este proyecto sólo se aplicarán técnicas de ML por lo que no se profundizará mucho en el DL. En la Ilustración 6Ilustración 6 se puede ver una imagen que muestra el esquema que se acaba de explicar.

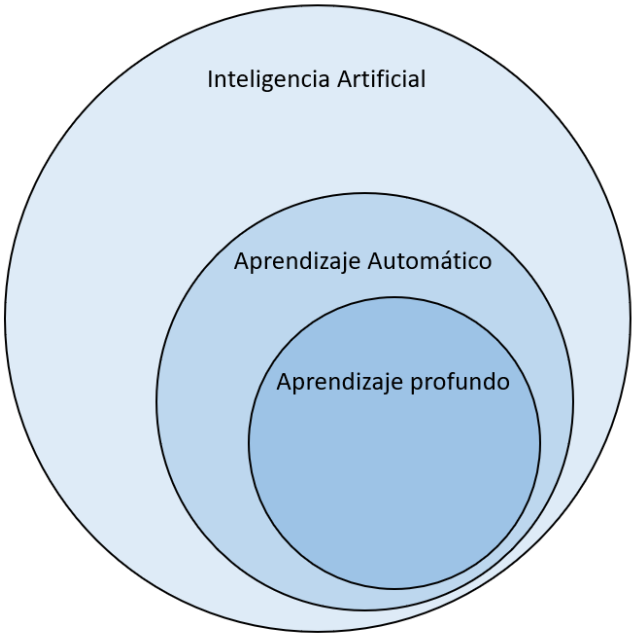


Ilustración 6. Esquema de Inteligencia artificia. Fuente: Elaboración propia.

Estos conceptos se componen de algoritmos de IA con el fin de crear sistemas expertos que hagan predicciones en función de una serie de datos de entrada.

Según el tipo de datos de los que se dispone y el problema que se plantea resolver, se trabaja con unos algoritmos u otros. Estos algoritmos se agrupan según el tipo de aprendizaje que reciben. Así, algunos de los tipos de aprendizajes más comunes son los siguientes:

2.4.1. Aprendizaje Supervisado

Este método utiliza conjuntos de datos etiquetados para entrenar los algoritmos [55], es decir, en el conjunto de datos de entrenamiento se indica la respuesta que se esperada. A medida que se van introduciendo datos de entrada, el algoritmo va modificando sus pesos hasta que el modelo se ajuste de forma correcta, intentando siempre evitar el sobreajuste o el desajuste. Algunos problemas que se pueden resolver con este tipo de

aprendizaje son clasificación de imágenes o predicciones de valores a lo largo del tiempo. Los algoritmos que se utilizarán para resolver el problema que se plantea en este proyecto utilizarán este tipo de aprendizaje. Los principales algoritmos utilizados en este tipo de aprendizaje son: regresión lineal, *Support Vector Machine* o árboles de decisión, sobre todos ellos se profundizará más adelante.

Dentro de aprendizaje supervisado, hay una subcategoría denominada aprendizaje perezoso.

2.4.1.1. Aprendizaje perezoso

Los algoritmos de ML que se clasifican como aprendizaje perezoso o aprendizaje basado en memoria funcionan guardando en memoria el conjunto de datos de entrenamiento para luego generalizar a nuevas instancias en función de alguna medida de similitud [56]. Este método también se denomina basado en instancias porque construye la hipótesis a partir de los datos de entrenamiento. La complejidad de este tipo de algoritmos depende del tamaño de los datos de entrenamiento.

Las ventajas de tipo de algoritmos son las siguientes:

- Permite a los modelos realizar aproximaciones locales a la función de destino, en vez de estimar para todo el conjunto de instancias.
- Debido a que guardan en memoria el conjunto de datos de entrenamiento y realizan los cálculos a predecir, los algoritmos se pueden adaptar muy fácilmente a nuevos datos.

Las desventajas de este tipo de algoritmos son las siguientes:

- Son algoritmos que consumen muchos recursos.
- Requieren una gran cantidad de memoria para almacenar los datos, además que cada consulta que se realice implica iniciar la identificación desde cero.

Algunos ejemplos de algoritmos basados en aprendizaje perezoso son: *K-Nearest-Neighbors*, mapa autoorganizado, cuantificación de vectores de aprendizaje y aprendizaje ponderado localmente.

2.4.2. Aprendizaje no supervisado

Se dice que un algoritmo de ML es de aprendizaje no supervisado cuando el algoritmo es utilizado para analizar y agrupar conjuntos de datos sin etiquetar [57]. Este tipo de aprendizaje se utiliza para descubrir agrupaciones de datos o patrones ocultos sin necesidad de ninguna intervención humana. El ejemplo del caso donde este tipo de

aprendizaje puede ser más útil es en el análisis exploratorio de datos, debido a su capacidad descubrir diferencias y similitudes en la información. Otros casos de uso pueden ser para estrategias de venta, segmentación de clientes o incluso la reducción del número de características de un modelo de ML mediante los procesos de reducción de dimensionalidad, análisis de componentes principales y descomposición del valor singular. Algunos ejemplos de algoritmos que utilizan este aprendizaje son las redes neuronales o el *clustering*.

2.4.3. Aprendizaje semisupervisado

El aprendizaje semisupervisado se presenta como un punto intermedio entre el aprendizaje supervisado y no supervisado [58]. Aquí, en el entrenamiento se utiliza un conjunto más pequeño de datos etiquetados para guiar la extracción de características de un conjunto de datos sin etiquetar de mayor tamaño. El aprendizaje semisupervisado es muy útil en algunas ocasiones ya que permite resolver el problema de no tener un conjunto de datos etiquetados lo suficientemente grande para entrenar un modelo de aprendizaje supervisado.

2.4.4. Aprendizaje en conjunto

El aprendizaje en conjunto consiste en entrenar varios modelos de ML, denominados modelos base o algoritmos simples, para después combinar sus resultados con el fin de que el modelo resultante sea capaz de realizar predicciones más precisas [59]. Los algoritmos que utilizan este método de aprendizaje se denominan algoritmos ensamblados. Los modelos obtenidos de algoritmos ensamblados suelen ser más fiables que los modelos individuales, por lo que suelen ser utilizados a un nivel más profesional.

Existen muchas técnicas para generar este tipo de modelos, desde técnicas muy simples como promediar los resultados de varios modelos distintos hasta técnicas desarrolladas específicamente para combinar las predicciones de muchos modelos base. Dos de los métodos más comunes para generar algoritmos ensamblados son:

2.4.4.1. Métodos de bagging

Los métodos de *bagging* utilizan los algoritmos simples en paralelo [60]. El objetivo de estos métodos es aprovechar la independencia que hay entre algoritmos simples, dado que, si varios modelos base llegan a la misma conclusión, se puede reducir el error promediando sus salidas. En la Ilustración 7 se puede ver cómo funciona la selección de un resultado a través de un método

bagging, en la que al haber una mayoría que elige la salida verde, la salida del modelo ensamblado es verde.

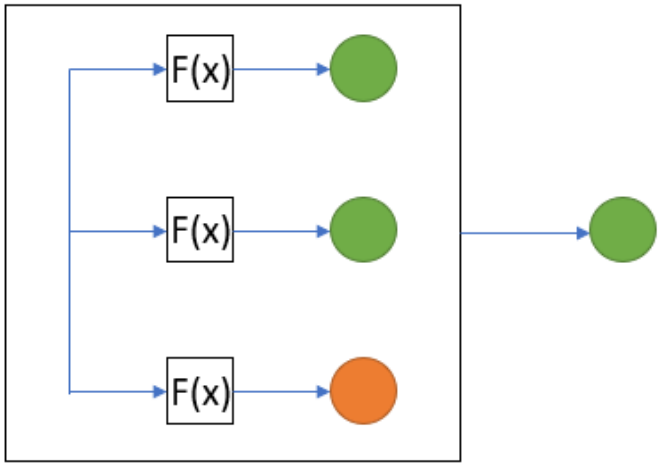


Ilustración 7. Funcionamiento de los métodos bagging. Fuente: Elaboración propia

Los métodos *bagging* nacen en 1996 por Leo Breiman del concepto de agregación de *Bootstrap* [61]. Son muy útiles por su forma de reducir la varianza de las predicciones haciendo cálculos más sencillos, como la votación para problemas de clasificación o el promedio para métodos de regresión.

Random Forest es un ejemplo de algoritmo que utiliza estos métodos para generar sus predicciones.

2.4.4.2. Métodos de boosting

En los métodos de *boosting*, los algoritmos simples son utilizados de forma continuada [62]. El objetivo de los algoritmos que utilizan estos métodos es reducir el sesgo de las predicciones, ya que al depender de los resultados de un modelo base anterior, se puede mejorar el rendimiento generando un modelo base posterior que dé más importancia a los errores cometidos. En la Ilustración 8 se puede ver un esquema del funcionamiento de los métodos *boosting*, en el que se ve cómo va variando la predicción a medida que va pasando por diferentes algoritmos:

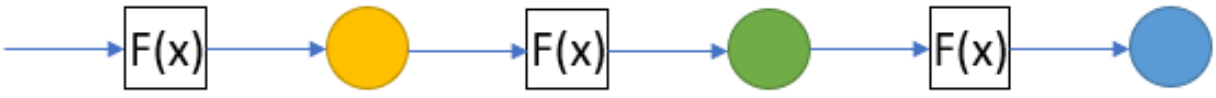


Ilustración 8. Funcionamiento de métodos boosting. Fuente: Elaboración propia

Las formas de realizar predicciones varían según el tipo de problema que se presente. Para problemas de clasificación se combinan las predicciones por medio de una votación, mientras que en problemas de regresión estas se combinan mediante una

suma ponderada. Un ejemplo de algoritmo que funciona con estos métodos es *AdaBoost*.

También es posible combinar los dos métodos para formar un algoritmo ensamblado aún más completo. *XGBoost* es un ejemplo de algoritmo que utiliza los dos métodos para generar predicciones.

Tal y como se ha visto, estos algoritmos son muy interesantes, ya que permiten disminuir el error producido por los modelos al realizar predicciones, dado que, al examinar los errores, estos estarán dispersos, por lo que se anulan entre sí. Mientras que las predicciones correctas se agrupan en torno a la verdadera opción, por lo que la predicción es mucho más precisa y fiable.

2.4.5. Aprendizaje automático

Como se ha visto antes, el aprendizaje automático es una subcategoría de la IA que intenta crear un modelo que sea capaz de aprender por sí mismo a resolver un problema sin haber sido programado para ello [63]. Así, se puede decir que ML se centra en el uso de datos y algoritmos para imitar la forma en la que aprenden los seres humanos, reconociendo patrones y su forma de actuar.

Se pueden diferenciar dos tipos de ML.

- ML profundo: El ML profundo, también conocido como Deep Learning es capaz de aprovechar conjuntos de datos para enseñar al algoritmo cómo actuar, este proceso se denomina entrenamiento. Los datos con los que se entrena el algoritmo pueden estar etiquetados o no, dependiendo de esto se trata de un aprendizaje supervisado o no supervisado. Este tipo de modelos son capaces de determinar por sí solos el conjunto de características que diferencian las categorías de datos. Este tipo de ML es capaz de trabajar sin intervención humana para procesar los datos, por lo que permite escalar en otras ramas. Dentro de esta categoría de ML se encuentran otras técnicas como el DL. Este es capaz de automatizar la mayoría de la extracción de características del proceso, eliminando gran parte de la intervención humana manual necesario y permitiendo el uso de conjuntos de datos más grandes.
- ML clásico: El ML clásico o no profundo depende más de la intervención humana para aprender que el ML profundo. Aquí se pasa al algoritmo un conjunto de parámetros necesarios para que este sea capaz de diferenciar los datos de entrada. Por lo general este tipo de ML requiere de más datos estructurados para aprender.

Para explicar el funcionamiento del ML se va a utilizar la explicación que da la escuela online *Berkeley Escuela de Información* [64]. Esta escuela divide el funcionamiento del ML en tres puntos principales:

1. Un proceso de decisión: El algoritmo lee los datos y realiza una serie de cálculos u otros pasos para predecir qué tipo de patrón busca encontrar su algoritmo.
2. Una función de error: Se utiliza la función de error como método para medir lo acertado de la predicción comparándola con ejemplos conocidos, siempre que estén disponibles. De esta forma se cuantifica el fallo del algoritmo.
3. Un proceso de actualización: Después de saber el fallo, el algoritmo actualiza una parte de su proceso de decisión para obtener una predicción más precisa.

Este proceso se repite una y otra vez hasta que el modelo producido es capaz de realizar predicciones con muy poco error. Este proceso es igual con todos los métodos de aprendizaje que se pueden dar.

A continuación, se verán más a fondo los algoritmos de ML que se utilizarán en este proyecto.

2.4.5.1. Regresión lineal

La regresión lineal es una técnica utilizada en estadística para predecir la relación entre dos o más variables [65]. Esta técnica ayuda a comprender el comportamiento de sistemas complejos para poder analizar con mayor facilidad los datos de los que están formados.

Gracias a la regresión lineal se puede crear un modelo lineal que describa la relación entre una variable independiente “y” como una función de una o varias variables independientes “X”. En la ecuación (11) se puede ver la ecuación general de un modelo de regresión lineal:

$$y = \beta_0 + \sum \beta_i x_i + \epsilon_i \tag{11}$$

Donde:

- y: Valor predicho de la variable dependiente
- β_0: Intersección con y
- $\beta_i x_i$: Coeficiente de regresión de la variable independiente.
- ϵ_i: Términos de error.

Sabiendo esto, se pueden definir varios tipos de regresión lineal.

Regresión lineal simple: Se habla de regresión lineal simple cuando el modelo utiliza solo una variable independiente. En la Ilustración 9 se puede ver un ejemplo de regresión lineal simple:

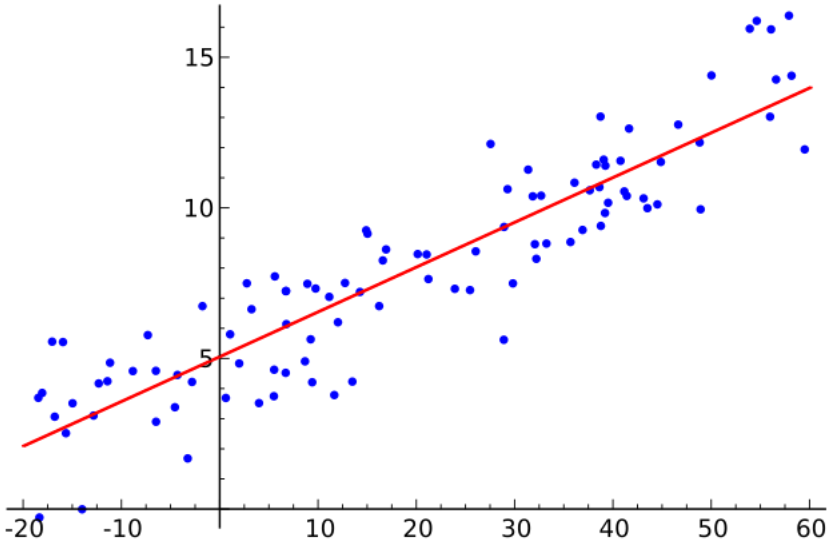


Ilustración 9. Ejemplo de regresión lineal. Fuente: [66]

- Regresión lineal múltiple: Se habla de regresión lineal múltiple cuando el modelo utiliza varias variables dependientes.
- Regresión lineal multivariante: Se habla de regresión lineal multivariante cuando el modelo se utiliza para varias variables dependientes.
- Regresión lineal múltiple multivariante: Se habla de regresión lineal múltiple multivariante cuando el modelo utiliza varias variables independientes para predecir varias variables dependientes.

2.4.5.2. Support Vector Machine

Support Vector Machine (SVM) es un algoritmo de ML muy utilizado, su objetivo es construir un hiperplano o un conjunto de estos en un espacio N-dimensional, donde N es el número de clases del conjunto de datos, que sea capaz de separar las clases de la mejor forma posible [67] [68]. En la Ilustración 10 se puede ver un ejemplo visual de cómo funciona la separación que crea SVM:

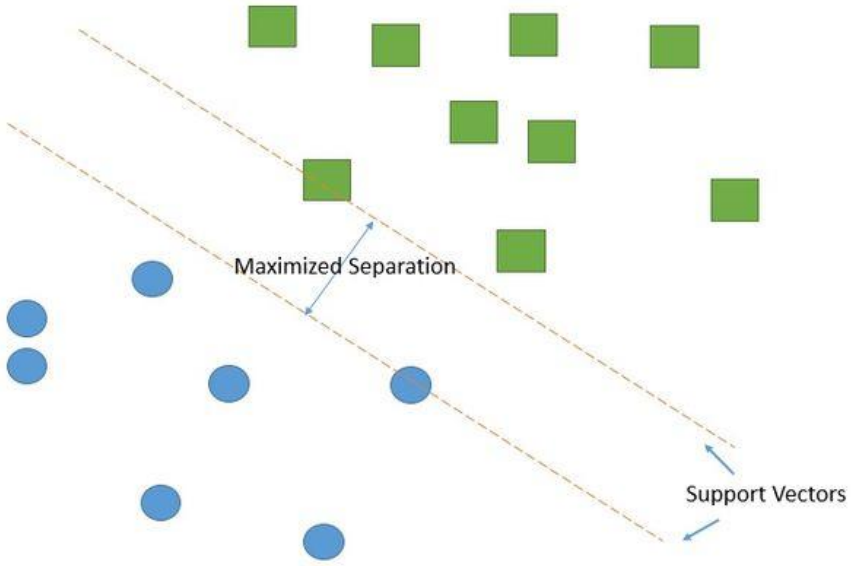


Ilustración 10. Ejemplo de SVM. Fuente: [69]

Para conseguir una buena separación, el hiperplano debe tener la mayor distancia al conjunto de datos de entrenamiento más cercanos de cualquier clase, este espacio se denomina margen funcional. Es mejor que este margen sea grande ya que cuanto más grande sea, menor será el error de generalización del modelo.

El algoritmo solo puede encontrar este hiperplano en problemas que permiten separación lineal, por lo que en la mayoría de los casos maximiza el margen permitiendo muchos valores erróneos. Aun así, por norma general, cuando el problema no es linealmente separable, los vectores de soporte son las muestras dentro de los límites del margen.

SVM se puede utilizar tanto para problemas de regresión, como de clasificación, para este proyecto se va a resolver un problema de regresión, por lo que se profundizará más en cómo funciona este algoritmo en este tipo de problemas.

Para problemas de regresión, se utiliza el método *Support Vector Regression* (SVR).

El modelo que produce el método SVR depende solo de un subconjunto de los datos de entrenamiento, esto es porque la función de coste ignora las muestras que realizan una predicción cerca de su objetivo.

Existen tres implementaciones diferentes de este método: SVR, NuSVR y LinearSVR; siendo SVR una aplicación más estándar, LinearSVR funciona proporcionando una implementación más rápida, pero teniendo en cuenta solo un kernel lineal, mientras que NuSVR implementa una formulación diferente. Desde la ecuación (12) hasta la (17), se explicará más detalladamente el funcionamiento matemático de este método:

Dado el conjunto de vectores de entrenamiento que se muestra en la ecuación (12):

$$x_i \in R^p, i = 1, 2, \dots, n \text{ y un vector } y \in R^n \tag{12}$$

Utilizando la ecuación (13) SVR permite resolver el siguiente problema:

$$\begin{gathered} \min_{w,b,\zeta,\zeta^*} \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^*) \\ sujeto\ a\ \mathrm{y_i} - \mathrm{w^T}\phi(\mathrm{x_i}) - \mathrm{b} \leq \varepsilon + \zeta_\mathrm{i}, \\ w^T \phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*, \\ \zeta_i, \zeta_i^* \geq 0, i = 1, \dots, n \end{gathered} \tag{13}$$

La ecuación (13) explica cómo se penalizan las muestras cuya predicción se aleja al menos ε de su objetivo. Estas muestras penalizan el objetivo en ζ_i o ζ_i^*, dependiendo de si sus predicciones están por encima o debajo de ε.

El problema doble se representa en la ecuación (14):

$$\begin{gathered} \min_{\propto,\propto^*} \frac{1}{2} (\propto - \propto^*)^T Q (\propto - \propto^*) + \varepsilon e^T (\alpha + \alpha^*) - y^T (\alpha - \alpha^*) \\ sujeto\ a\ e^T (\alpha - \alpha^*) = 0 \\ 0 \leq \alpha_i, \alpha_i^* \leq C, i = 1, \dots, n \end{gathered} \tag{14}$$

En la ecuación (14) se puede ver que e es el vector de todos los unos, Q es una matriz semidefinida n x n, el kernel está definido por la ecuación (15):

$$Q_{ij} \equiv K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{15}$$

Aquí los vectores de entrenamiento se mapean implícitamente en un espacio dimensional superior mediante la función ϕ.

Así, la predicción la predicción que se realiza está descrita en la ecuación (16):

$$\sum_{i \in SV} (\alpha_i - \alpha_i^*) K(x_i, x) + b \tag{16}$$

Se puede acceder a los parámetros de la ecuación (16) utilizando los atributos, “dual_coef” para indicar la diferente entre $\alpha_i - \alpha_i^*$, “support_vectors” que guarda los vectores de soporte. Y "intercept” que guarda el término independiente b.

En LinearSVR, el principal problema se puede formular como se muestra en la ecuación (17):

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \max(0, |y_i - (w^T \phi(x_i) + b)| - \varepsilon) \quad (17)$$

Donde se hace uso de la pérdida insensible a épsilon, es decir, se ignoran los errores inferiores a ε. Esta es la forma que optimiza directamente LinearSVR.

Hay distintos tipos de SVM [67], estos son:

- Función de Base Radial (RBF) o Gaussiana: El kernel RBF se utiliza para el aprendizaje de una clase. Su función se muestra en la ecuación (18):

$$K(x_1, x_2) = \exp\left(-\frac{|x_1 - x_2|^2}{2\sigma^2}\right) \quad (18)$$

  Donde σ representa la anchura del kernel.
- Lineal: El kernel lineal se utiliza para para el aprendizaje de dos clases. Su función se muestra en la ecuación (19):

$$K(x_1, x_2) = x_1^T x_2 \quad (19)$$

  Sirve para el aprendizaje de dos clases.
- Polinómica: El kernel polinómico se utiliza para para el aprendizaje de dos clases. Su función se muestra en la ecuación (20):

$$K(x_1, x_2) = \left(x_1^T x_2 + 1\right)^{\rho} \quad (20)$$

  Donde ρ representa el orden del polinomio.
- Sigmoide: El kernel sigmoide Representa un kernel de Mercer solo para determinados valores β_0 y β_1. Su función se muestra en la ecuación (21):

$$K(x_1, x_2) = \tanh\left(\beta_0 x_1^T x_2 + \beta_1\right) \quad (21)$$

2.4.5.3. K-Nearest Neighbors

KNN es un algoritmo de aprendizaje supervisado, que se basa en analizar la proximidad de otros puntos de datos para predecir el valor de un punto individual [70] [71]. En la Ilustración 11 se puede ver un ejemplo visual del funcionamiento de KNN:

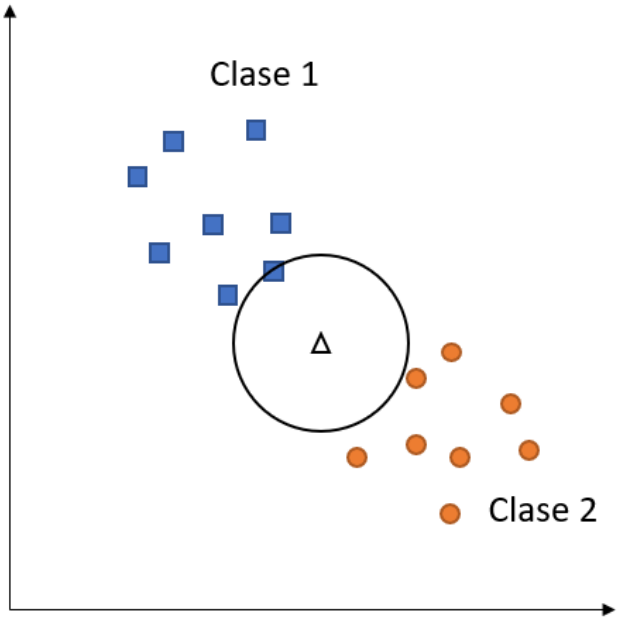


Ilustración 11. Ejemplo de KNN. Fuente: Elaboración propia

Este algoritmo es capaz de resolver tanto problemas de regresión como de clasificación, aunque su uso es más común en problemas de clasificación. En este tipo de problemas, el algoritmo analiza los puntos que tiene más cercanos, y en base a estos, clasifica el punto que se desea predecir, dándole a este el valor de la mayoría de los otros puntos.

En problemas de regresión el algoritmo varía un poco, dado que ya no trabaja con valores discretos, sino con continuos, por lo que ahora el valor predicho pasa a tener el valor medio de los otros puntos. Para los dos tipos de problemas hay que definir la distancia con la que se miden los puntos ya que según la distancia con la que se miden estos, varían el número de puntos que influyen en la predicción.

También hay que destacar que el algoritmo KNN entra dentro del grupo de modelos de aprendizaje perezoso, por lo que solo almacena el conjunto de datos de entrenamiento en vez de entrenar el modelo en sí, por lo que el cálculo se produce al realizar la predicción sobre el punto. Este es uno de los algoritmos más básicos debido a su sencillez y precisión. Sin embargo, este algoritmo se vuelve muy ineficiente a medida que el conjunto de datos va creciendo. Algunos de los usos más comunes de este algoritmo son sistemas de recomendación, reconocimiento de patrones, minería de datos, predicciones de mercados financieros, etc.

Para que entender correctamente el funcionamiento de este algoritmo, hay que tener en cuenta las siguientes dos variables:

- Métricas de distancia: Para determinar qué puntos están próximos al punto que se desea consultar, primero hay que calcular la distancia entre este punto y los demás. Las métricas de distancia ayudan a formar los límites de decisión, que dividen los puntos de consulta en regiones distintas. Los límites de decisión se suelen visualizar mediante diagramas de *Voronoi*.

  Las distancias más usadas son la distancia euclídea, Manhattan y Minkowski, siendo la distancia euclídea la más común de usar.
- K: El valor K define cuantos puntos se van a comprobar para determinar el valor de un punto específico. Según el valor que se asigne a K puede llevar a un sobreajuste o ajuste insuficiente. Los valores más bajos de K pueden tener una varianza alta, pero con un sesgo más bajo, mientras que los valores altos de K pueden producir una varianza más baja, pero con un sesgo más alto. La elección de este valor dependerá de los datos de entrada, ya que, si el conjunto de datos dispone de más valores atípicos o con mucho ruido, el modelo seguramente funcione mejor con valores de K más altos. Por lo general, es mejor tener un número impar de K para evitar empates en la clasificación. También es recomendable utilizar técnicas de validación cruzada para elegir un valor de K óptimo para el conjunto de datos.

Con todo esto, es posible identificar algunas ventajas y desventajas de este algoritmo:

- Ventajas:
  - Se adapta fácilmente: Debido a que el conjunto de datos de entrenamiento se almacena en memoria y el cálculo se realiza al hacer la predicción, es posible que el modelo se ajuste al añadir nuevos datos.
  - Pocos hiper-parámetros: Tal y como se ha visto antes, solo es necesario dar un valor a K y proporcionar una métrica de distancia para que este algoritmo funcione.
  - Fácil de implementar: Debido a los dos puntos anteriores, este algoritmo se convierte en uno realmente fácil de implementar y uno de los más fáciles de aprender.

- Desventajas:
  - No escala bien: Al ser un algoritmo de aprendizaje perezoso, ocupa más memoria que otros algoritmos. Esto hace que sea un algoritmo muy costoso para conjuntos de datos grandes. Aunque gracias a algunas estructuras de datos como *Ball-Tree*, esta desventaja puede pasar un poco más desapercibida.

- Maldición de dimensionalidad: El algoritmo no funciona bien con entradas de datos de alta dimensión.

  Este fenómeno se conoce como fenómeno de pico, el cual consiste en que después de haber alcanzado el número óptimo de características, las características adicionales aumentan la cantidad de errores de la predicción, especialmente si se cuenta con un conjunto de datos más pequeño.
- Es propenso a sobreajuste: Debido al punto anterior, El algoritmo también es más propenso al sobreajuste. Para evitar esto se utilizan técnicas de selección de características y de reducción de dimensionalidad, el valor de k también puede influir en el comportamiento del modelo, ya que como hemos visto antes, un valor de k mal escogido puede dar lugar a un modelo mal ajustado

2.4.5.4. Random Forest

Random Forest es un algoritmo registrado por Leo Breiman y Adele Cutler [72]. Este algoritmo está formado de un conjunto de árboles de decisión, combinando sus salidas para obtener un mejor resultado. Al ser un algoritmo fácil de usar y muy flexible, además de capaz de gestionar tanto problemas de regresión como de clasificación, es uno de los algoritmos de ML más utilizados.

Para entender correctamente el funcionamiento de *Random Forest* primero es necesario saber qué son los árboles de decisión y entender su funcionamiento.

2.4.5.4.1. Árboles de decisión

Los árboles de decisión componen el algoritmo *Random Forest*. Estos parten de una pregunta sencilla y se van continuando preguntas hasta llegar a una respuesta [73]. Estas preguntas son los nodos de decisión, y cada posible respuesta actúa como la rama del árbol.

Así, con cada nodo se ayuda a tomar una decisión final, la cual se denomina el nodo hoja. El objetivo de los árboles de decisión es encontrar la mejor forma de división para agrupar los datos, y estos son entrenados a partir del algoritmo de Árbol de Clasificación y Regresión (CART). Para evaluar la calidad de la división se utilizan métricas como impureza Gini, ganancia de información o error medio cuadrático. En la Ilustración 12 se puede ver un esquema de la composición de un árbol de decisión.

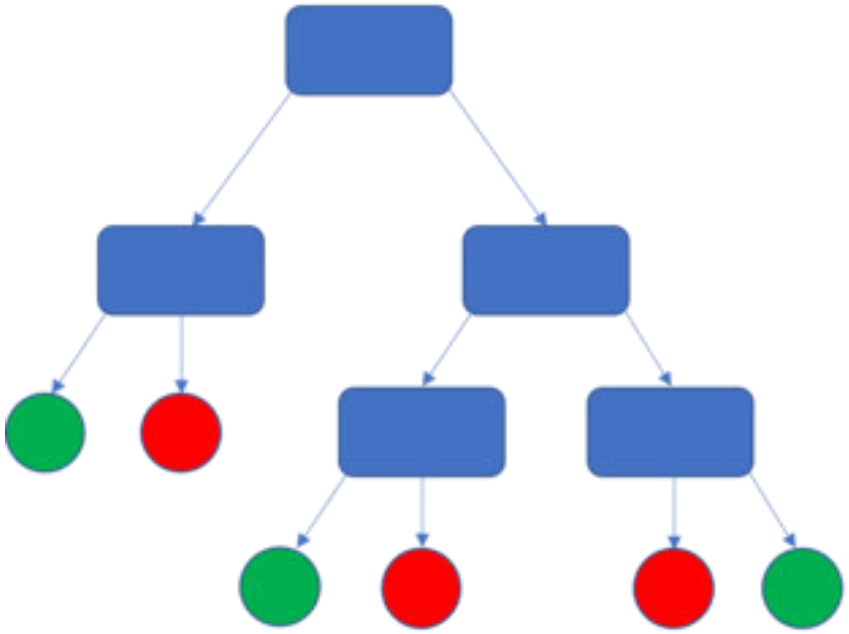


Ilustración 12. Ejemplo de un árbol de decisión. Fuente: [74]

Aunque los árboles de decisión son muy usados en aprendizaje supervisado, son muy propensos a problemas como el sesgo y el sobreajuste. Por eso en el algoritmo *Random Forest* se tienen en cuenta varios árboles de decisión que no estén correlacionados entre sí, ya que así son capaces de predecir resultados más precisos.

2.4.5.4.2. Algoritmo Random Forest

El algoritmo Random Forest [72] utiliza El aprendizaje en conjunto consiste en entrenar varios modelos de ML, denominados modelos base o algoritmos simples, para después combinar sus resultados con el fin de que el modelo resultante sea capaz de realizar predicciones más precisas [59]. Los algoritmos que utilizan este método de aprendizaje se denominan algoritmos ensamblados. Los modelos obtenidos de algoritmos ensamblados suelen ser más fiables que los modelos individuales, por lo que suelen ser utilizados a un nivel más profesional.

Existen muchas técnicas para generar este tipo de modelos, desde técnicas muy simples como promediar los resultados de varios modelos distintos hasta técnicas desarrolladas específicamente para combinar las predicciones de muchos modelos base. Dos de los métodos más comunes para generar algoritmos ensamblados son:

Métodos de bagging para generar predicciones, pero actúa como una extensión de estos, ya que además de aplicar métodos de *bagging*, suma otro método de aleatoriedad de características o *feature bagging*, lo que permite generar un subconjunto aleatorio que garantiza una baja correlación entre los árboles de decisión.

Esta es una diferencia importante del algoritmo *Random Forest* con el algoritmo de árboles de decisión, ya que mientras que este último considera todas las divisiones posibles de características, *Random Forest* solo selecciona un subconjunto de ellas. Esto produce que se reduzca el riesgo de sobreajuste, el sesgo y la varianza general, lo que permite que se puedan realizar predicciones más precisas. En la Ilustración 13 se puede ver un ejemplo de la composición del algoritmo *Random Forest*, teniendo en su interior varios árboles de decisión y con dos salidas al final:

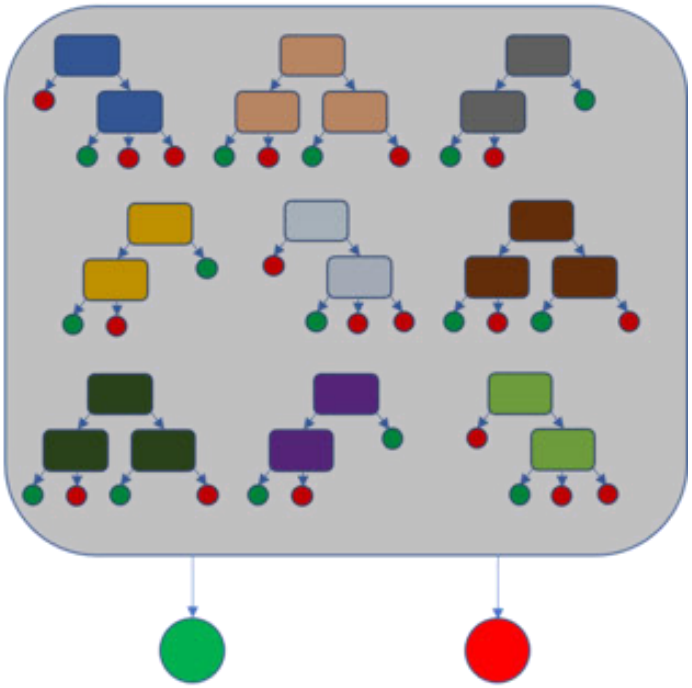


Ilustración 13. Ejemplo de Random Forest. Fuente: [74]

El algoritmo *Random Forest* tiene muchos hiperparámetros, pero tiene tres que son especialmente importantes, y deben establecerse antes del entrenamiento. Estos son el tamaño de los nodos, el número de árboles y el número de características muestreadas. A partir de aquí, el algoritmo se puede utilizar para resolver problemas tanto de regresión como de clasificación.

Para entender el funcionamiento de este algoritmo hay que saber que se compone de una colección de árboles de decisión y a su vez, cada árbol del conjunto se compone de una muestra de datos extraída de un conjunto de entrenamiento con reemplazo, denominado muestra Bootstrap. De aquí, se reserva un tercio como datos de prueba, lo que se conoce como *Out Of Bag* (OOB). Posteriormente, se añade otra instancia de aleatoriedad a través del *feature bagging*, esto añade más diversidad al conjunto de datos y reduce la correlación entre los árboles de decisión. La determinación de la predicción varía según el tipo de problema. Para una tarea de regresión, se promediarán

los árboles de decisión individuales, y para una tarea de clasificación, se tendrá en cuenta la variable categórica más frecuente para dar como resultado la clase predicha. Por último, se utiliza la muestra OOB para la validación cruzada.

Este algoritmo presenta una serie de ventajas y desventajas.

- Ventajas:
    - Reducción del riesgo de sobreajuste: Como hemos visto anteriormente, al no estar los árboles correlacionados entre sí, es menos probable que el modelo final esté sobre ajustado, ya que el promedio de árboles no correlacionados reduce la varianza global y el error de predicción.
    - Proporciona flexibilidad: Es un algoritmo que puede resolver problemas tanto de regresión como de clasificación. Además, el *feature bagging* permite estimar valores faltantes, ya que mantiene la precisión cuando faltan una parte de los datos.
    - Facilidad para determinar la importancia de las características: El algoritmo facilita la evaluación de la importancia de las variables al modelo. Hay varias formas de evaluarlo, como la importancia de Gini o la disminución media de la impureza, también llamada *Mean Decrease in Impurity* (MDI), que suelen utilizarse para medir cuanto disminuye la precisión del modelo al excluir una determinada variable. Otra forma de evaluar muy importante es la importancia de permutación, también conocida como precisión de disminución media o *Mean Decrease Accuracy* (MDA), este método identifica la disminución media de la precisión al permutar aleatoriamente los valores de las características en las muestras OOB.
- Desventajas:
    - Proceso lento: Los procesos de entrenamiento permiten manejar grandes conjuntos de datos y proporcionar predicciones más precisas, pero estos pueden llegar a ser muy lentos ya que aplican el conjunto de datos a cada árbol de decisión individual.
    - Requiere muchos recursos: Ya que este algoritmo está preparado para manejar conjuntos de datos más grandes, necesita más recursos para almacenarlos.

- Es más complejo: Es más fácil de interpretar la predicción de un único árbol que la de un bosque entero de estos

2.4.5.5. Descenso de gradiente

El descenso de gradiente [75] es un algoritmo de optimización muy usado para entrenar modelos de ML y redes neuronales. En este algoritmo, la función de coste calibra su precisión con cada actualización de parámetros. Hasta que la función no es igual o cercana a cero, el modelo se seguirá ajustando hasta alcanzar el menor error posible.

El funcionamiento del descenso de gradiente es parecido al de la regresión lineal, ya que en esta última se deben encontrar los parámetros m y b de la función descrita por la ecuación (22) que permitan que la recta obtenida se ajuste lo mejor posible a los datos:

$$y = mx + b \tag{22}$$

Esto requiere calcular el error entre los datos reales y los predichos, utilizando la fórmula del error medio cuadrático. El algoritmo de descenso de gradiente funciona similar, pero está basado en una función convexa como la dada a continuación:

El punto inicial se escoge de forma arbitraria, a partir de aquí se obtiene la pendiente a través de la derivada y utilizamos una línea tangente a esta para ver los pasos de la pendiente. La pendiente da la información sobre las actualizaciones de parámetros. Al principio la pendiente da pasos grandes, pero estos se van reduciendo con cada iteración hasta que encuentra el punto más bajo de la curva, conocido como el punto de convergencia.

El objetivo de este algoritmo es minimizar la función de coste o los residuos. Para esto se requieren dos cosas, una dirección y una tasa de aprendizaje. Estos factores determinan como se va a comportar el algoritmo a lo largo de las iteraciones, permitiéndolo así encontrar un mínimo local o global de la función.

- Tasa de aprendizaje: Indica el tamaño de los pasos que son dados para alcanzar el mínimo. Se suelen utilizar valores bajos y se van actualizando según los valores de la función de coste. Utilizar valores altos puede ayudar a agilizar el proceso, pero es probable que nunca se llegue a alcanzar un mínimo; mientras que utilizar valores muy bajos hace más probable encontrar un mínimo, pero al tener que dar muchos pasos para encontrarlo se convierte en un algoritmo poco eficiente.
- Función de coste: La ecuación (23) define la función de coste:

$$\mathrm{f}(m,b) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - (mx_i + b)\right)^2 \tag{23}$$

Tal y como se muestra en la ecuación (24), derivando la función de coste descrita en la ecuación (23) se puede obtener el gradiente:

$$f'(m,b) = \begin{bmatrix} \frac{df}{dm} \\ \frac{df}{db} \end{bmatrix} = \begin{bmatrix} \frac{1}{N}\sum -2x_i\left(y_i - (mx_i + b)\right) \\ \frac{1}{N}\sum -2\left(y_i - (mx_i + b)\right) \end{bmatrix} \tag{24}$$

La función de coste mide el error entre los valores predichos y los valores reales en cada posición. Esto proporciona al modelo la capacidad de ver su fallo y ajustar los parámetros para minimizar el error y encontrar un mínimo local o global más fácilmente. Esto se itera varias veces moviéndose en la dirección que, de el paso hacia la zona más próxima a un mínimo, hasta que la función de coste sea próxima a cero.

> La función de coste y de pérdida comúnmente se consideran sinónimos, pero no son lo mismo. La función de pérdida se refiere al error de ejemplo de entrenamiento, mientras que la función de coste calcula el error medio a lo largo de todo el set de entrenamiento.

Aunque el descenso de gradiente sea de los algoritmos de optimización más utilizados, también presenta algunos problemas. Algunos son:

- Mínimos locales y puntos de silla: En problemas con una sola curva convexa, es fácil encontrar el mínimo global, pero en problemas que presentan varias curvas es muy probable caer en mínimos locales antes que, en el mínimo global, que es donde el modelo va a obtener mejores resultados.
  Este problema también puede suceder en un punto de silla, ya que aquí la derivada también es cero por lo que puede confundirlo con un mínimo de la función.
  Los gradientes ruidosos pueden ayudar a escapar de este tipo de problemas.

- Gradientes invisibles y explosivos: Este tipo de problemas son más fáciles de encontrar en redes neuronales más profundas, especialmente cuando se utilizan algoritmos de descenso de gradiente y de *backpropagation*.
  - Gradientes que desaparecen: Esto ocurre cuando el gradiente es muy pequeño. Al volver hacia atrás durante el *backpropagation*, el gradiente continúa haciéndose más pequeño, causando que las capas anteriores de la red neuronal aprendan más despacio que las últimas capas. Esto

hace que el peso de sus algoritmos se vuelva insignificante y que el algoritmo no siga aprendiendo.

- Gradientes explosivos: Este problema ocurre cuando el gradiente es muy largo, ya que se crea un modelo inestable. Aquí los pesos crecen hasta ser muy grandes, hasta que de repente presentan valor nulo (*NaN*). Una solución para esto es aprovechar una técnica de reducción de dimensionalidad, que permita ayudar a reducir la complejidad dentro del modelo.

Hay tres tipos de algoritmos de descenso de gradiente. Estos son:

- Descenso de gradiente por lotes: Este algoritmo suma el error de cada punto del set de entrenamiento, actualizando el modelo una vez todos los entrenamientos han sido evaluados.

  Este proceso es una época de entrenamiento.

  Esto permite una computación más eficiente, pero requiere más tiempo de procesamiento para datos de entrenamiento muy grandes y necesita más memoria para almacenar todos los lotes. Este algoritmo también puede presentar un error de gradiente estable y convergencia, pero ese punto de convergencia puede ser un mínimo local en vez de un mínimo global, por lo que no es el punto ideal.
- Descenso de gradiente estocástico (SGD): SGD corre una época de entrenamiento por cada ejemplo dentro del conjunto de datos, y actualiza cada parámetro de entrenamiento una vez. De esta forma solo necesitas mantener un ejemplo de entrenamiento en memoria, lo que lo hace más fácil de almacenar. Esto puede ofrecer más detalle y velocidad, pero también puede producir pérdidas de eficiencia computacional en comparación con el descenso de gradiente por lotes. Tantas actualizaciones pueden provocar gradientes ruidosos, pero esto a veces también puede ayudar a escapar de un mínimo local y encontrar un mínimo global.
- Descenso de gradiente por mini lotes: Este algoritmo combina conceptos de los dos tipos de descenso de gradiente vistos anteriormente (SGD y por lotes). Este algoritmo divide el set de entrenamiento en lotes pequeños y aplica actualizaciones en cada lote. Esta aproximación proporciona un balance entre la computación eficiente del descenso de gradiente por lotes y la velocidad del SGD.

2.4.5.5.1. Gradient Boosting Regressor

El algoritmo *Gradient Boosting* [76] es uno de los algoritmos más comunes ya que es un algoritmo potente que permite encontrar relaciones no lineales entre las variables dependientes y la variable independiente del modelo. Además, es muy fácil de usar ya que puede trabajar con valores nulos, atípicos y categóricos de alta cardinalidad. Hay bibliotecas muy populares que aplican este algoritmo como *XGBoost* o *LightGBM*.

Este algoritmo trabaja creando varios modelos débiles y combinándolos para crear uno con buen rendimiento.

Para ello parte de una predicción muy simple y, a partir de sus residuos, va creando árboles cada vez más complejos hasta tener uno que es capaz de predecir correctamente.

El funcionamiento matemático del algoritmo viene definido por las ecuaciones (25) hasta la (29):

1. La ecuación (25) muestra cómo se inicializa el modelo con un valor constante:

$$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, \gamma) \tag{25}$$

Aquí se inicia el modelo a partir de la función de pérdida L, y se queda solo con el valor mínimo, así el valor inicial es el más bajo de esta función.

2. Para $m = 1$ a M, siendo todos los procesos de este punto iteraciones, M representa el número de iteraciones y m representando el índice de cada árbol.
   2.1. La ecuación (26) muestra cómo se computan los residuos

$$r_{im} = -\left[\frac{\partial L\big(y_i, F(x_i)\big)}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \text{ para } i = 1, \dots, n \tag{26}$$

Para ello se deriva la función de pérdida. Así se obtienen los residuos del modelo. Esta técnica de usar el gradiente para minimizar la pérdida es muy similar al descenso de gradiente.

   2.2. Entrenar árboles de regresión con x capas contra r y crear nodos terminales racionados que se identificarán mediante la ecuación (27):

$$R_{jm}\ para\ j = 1, \dots, J_m \tag{27}$$

   2.3. La ecuación (28) muestra la siguiente función de pérdida a computar.

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma) \text{ for } j = 1, \ldots, J_m \quad (28)$$

Se está buscando una función que minimice la función de pérdida en cada nodo terminal. $x_i \in R_{jm}$ significa que se agrega la función de pérdida en cada valor de x que pertenece al nodo terminal R_{jm}. En resumen, esta función es los valores regulares predichos de árboles de regresión que son la media de los valores principales (residuos en este caso) en cada nodo terminal.

2.4. La ecuación (29) muestra la ecuación con la que se actualiza el modelo.

$$F_m(x) = F_{m-1}(x) + v \sum_{j=1}^{J_m} \gamma_{jm} 1(x \in R_{jm}) \quad (29)$$

En este último paso se actualiza la predicción del modelo combinado. $\gamma_{jm} 1(x \in R_{jm})$ significa que se escoje el valor yjm si una x dada cae en un nodo terminal Rjm. Como todos los nodos son exclusivos. Cualquier x válida es permite que se actualice la predicción.

V es la tasa de aprendizaje, su rango está entre 1 y 0 y controla el grado de contribución de un árbol de predicción adicional y a la predicción combinada Fm. Una tasa de aprendizaje baja reduce el efecto del árbol adicional, pero también reduce la probabilidad de que el modelo sufra sobreajuste.

Todos los pasos del punto dos se iteran M veces. Lo que significa añadir M árboles al modelo combinado.

2.4.5.5.2. SGD Regressor

La clase SGD *Regressor* [77] implementa una rutina de aprendizaje basada en el algoritmo de descenso de gradiente estocástico que se ha visto anteriormente. Esta clase es adecuada para problemas de regresión con una gran cantidad de muestras de entrenamiento (más de 10.000 muestras), para otros problemas es recomendable utilizar otros algoritmos como *Ridge*, *Lasso* o *ElasticNet*.

SGD *Regressor* admite diferentes funciones de perdida y penalizaciones.

Las funciones de pérdida se pueden configurar mediante el parámetro *loss*. Las funciones admitidas son las siguientes:

- *squared_loss*: Mínimos cuadrados ordinarios.
- *huber*: Pérdida de *huber* para regresión robusta.

- *Épsilon_insensitive*: Regresión de vectores de soporte lineal

Para ajustar las penalizaciones, hay que modificar el parámetro *penalty*, este determina la regularización que se utilizará.

El proceso SGD se puede describir matemáticamente como:

Dado un set de entrenamiento como el descrito en la ecuación (30):

$$(x_1, y_1), \dots, (x_n, y_n) \text{ donde } x_i \in R^m \text{ y } y_i \in \mathcal{R}(y_i \in -1, 1), \tag{30}$$

El objetivo es aprender una función lineal como la descrita en la ecuación (31):

$$f(x) = w^T x + b \tag{31}$$

Con los parámetros del modelo $w \in R^m$ y la intercepción $b \in R$. Para encontrar los parámetros del modelo, se minimiza el error del entrenamiento regularizado, este error se describe en la ecuación (32):

$$\mathrm{E(w, b)} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{L}\big(\mathrm{y_i}, \mathrm{f(x_i)}\big) + \alpha \mathrm{R(w)} \tag{32}$$

Donde L es la función de pérdida que mide el ajuste del modelo y R es un término de regularización (penalización) que penaliza la complejidad del modelo. $\mathrm{A} > 0$ es un hiperparámetro no negativo que controla la fuerza de regularización.

Según la elección de L, puede dar en diferentes regresores, algunos de ellos son:

- Perceptrón: La ecuación (33) muestra la función de pérdida utilizando un perceptrón.

$$\mathrm{L}\big(\mathrm{y_i}, \mathrm{f(x_i)}\big) = \max\big(0, -\mathrm{y_i f(x_i)}\big) \tag{33}$$

- Error cuadrático: La ecuación (34) muestra la función de pérdida utilizando el error cuadrático. El resultado es equivalente a una regresión lineal.

$$\mathrm{L}\big(\mathrm{y_i}, \mathrm{f(x_i)}\big) = \frac{1}{2}\big(\mathrm{y_i} - \mathrm{f(x_i)}\big)^2 \tag{34}$$

- Huber: El regresor Huber es menos sensible a valores atípicos que los mínimos cuadrados, salvo en el caso mostrado por la ecuación (35):

$$|y_i - \mathrm{f}(x_i)| \leq \varepsilon \text{ y } L\big(y_i, f(x_i)\big) = \varepsilon |y_i - f(x_i)| - \frac{1}{2}\varepsilon^2 \tag{35}$$

- Huber modificado: La ecuación (36) muestra la función de pérdida utilizando un Huber modificado:

$$\mathrm{L}(\mathrm{y_i}, \mathrm{f}(\mathrm{x_i})) = \max(0{,}1 - \mathrm{y_i f}(\mathrm{x_i}))^2 \text{ si } L(y_i, f(x_i)) = -4 y_i f(x_i) \quad (36)$$

- *Epsilon-Insensitive*: La ecuación (37) muestra la función de pérdida utilizando el error cuadrático. El resultado es equivalente al algoritmo *Support Vector Regression*.

$$L(y_i, f(x_i)) = \max(0, |y_i - f(x_i)| - \varepsilon) \quad (37)$$

Tal y como muestra la Ilustración 14, todas las funciones de pérdida pueden considerarse como un límite superior en el error de clasificación errónea (Pérdida cero-uno).

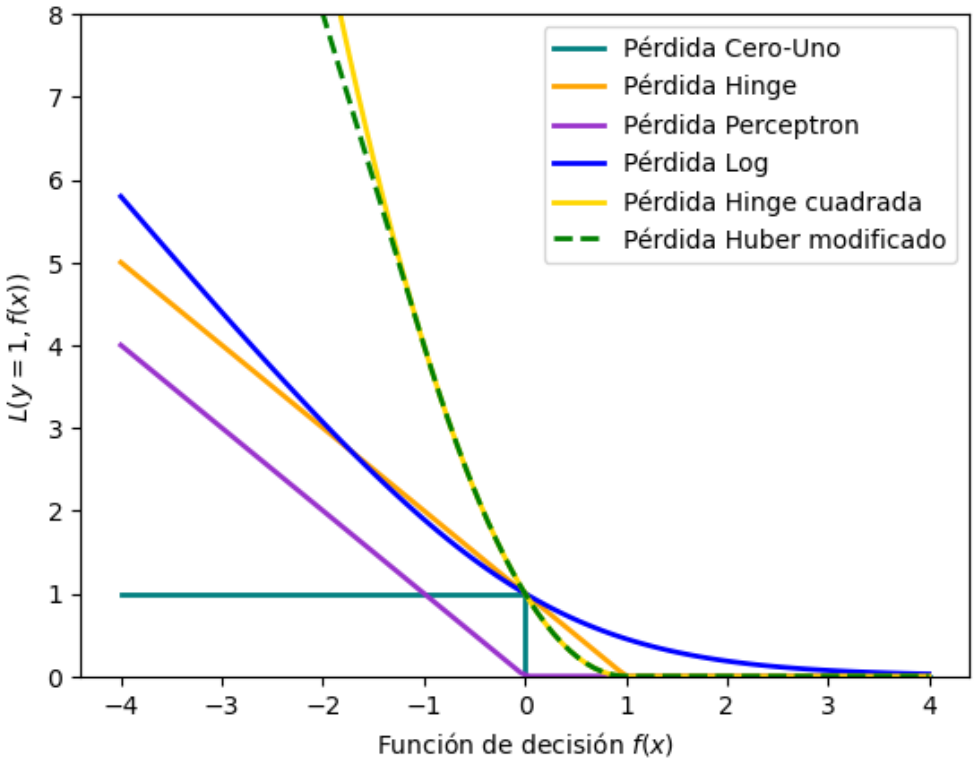


Ilustración 14. Función de pérdida según el regresor utilizado. Fuente: Elaboración propia

Algunas elecciones populares para la normalización del término R (el parámetro *penalty*) incluyen:

- *L2 norm*: Corresponde a Ridge. Su función de normalización se describe por la ecuación (38).

$$R(w) := \frac{1}{2}\sum_{j=1}^{m} w_j^2 = ||w||_2^2 \quad (38)$$

- *L1 norm*: Corresponde a Lasso. Su función de normalización se describe por la ecuación (39).

$$R(w) \coloneqq \sum_{j=1}^{m} |w_j| \tag{39}$$

que lleva a soluciones dispersas.

- *Elastic Net*: Su función de normalización se describe por la ecuación (40).

$$R(w) \coloneqq \frac{\rho}{2} \sum_{j=1}^{n} w_j^2 + (1-\rho) \sum_{j=1}^{m} |w_j| \tag{40}$$

Elastic Net es una combinación convexa de L2 y L1, donde ρ es dada por 1 – *l1_ratio*, siendo *l1_ratio* un parámetro de la clase.

La Ilustración 15 muestra los diferentes términos de regularización:

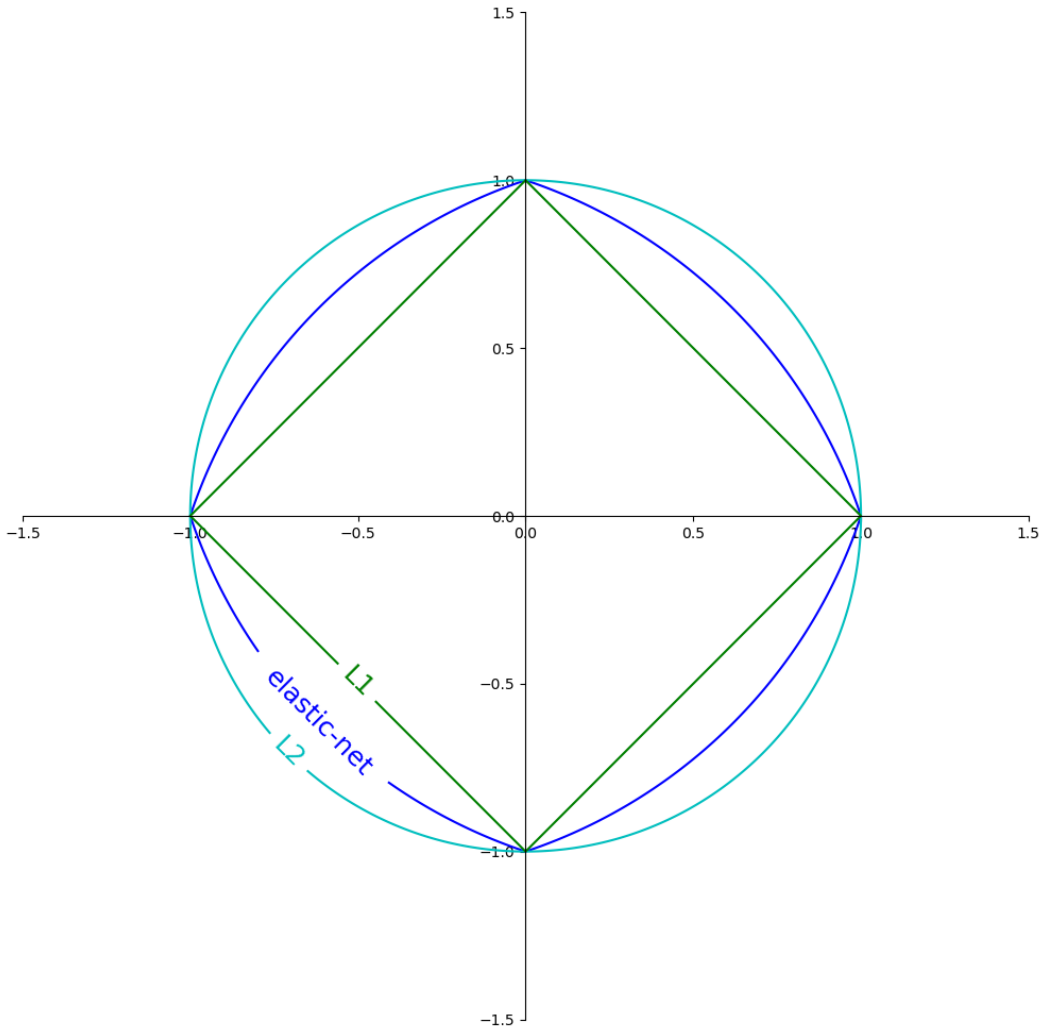


Ilustración 15. Representación de los diferentes términos de regularización. Fuente: Elaboración propia

2.4.5.6. AdaBoost

El algoritmo AdaBoost [78], también llamado Adaptative Boosting, es una técnica de Machine Learning utilizado como un método de ensamblaje. El algoritmo más utilizado con AdaBoost es el de árboles de decisión con un nivel, lo que quiere decir que los árboles de decisión solo generan una división. Estos árboles se denominan tocones de decisión. La Ilustración 16 muestra un ejemplo de un tocón de decisión.

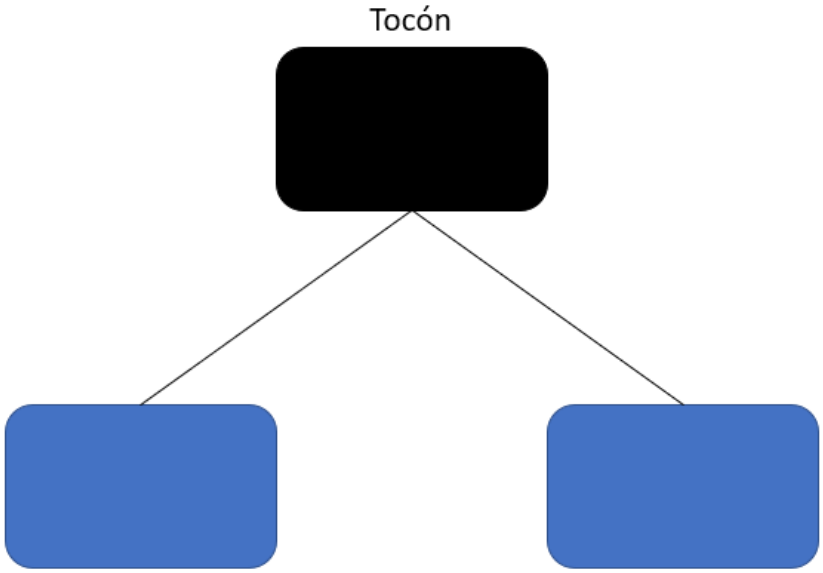


Ilustración 16. Ejemplo de tocón de decisión. Fuente: Elaboración propia

Lo que hace el algoritmo es construir un modelo y ponderar igual todos los puntos de datos. Después realiza una clasificación y vuelve a ponderar los puntos, dando esta vez más peso a los puntos que están mal clasificados, así se les da más importancia a estos en futuras iteraciones. Este proceso se repite hasta que el error sea muy bajo. La Ilustración 17 muestra un ejemplo del funcionamiento del algoritmo AdaBoost, en el que se puede ver como todos los modelos base aportan sus pesos al algoritmo ensamblado.

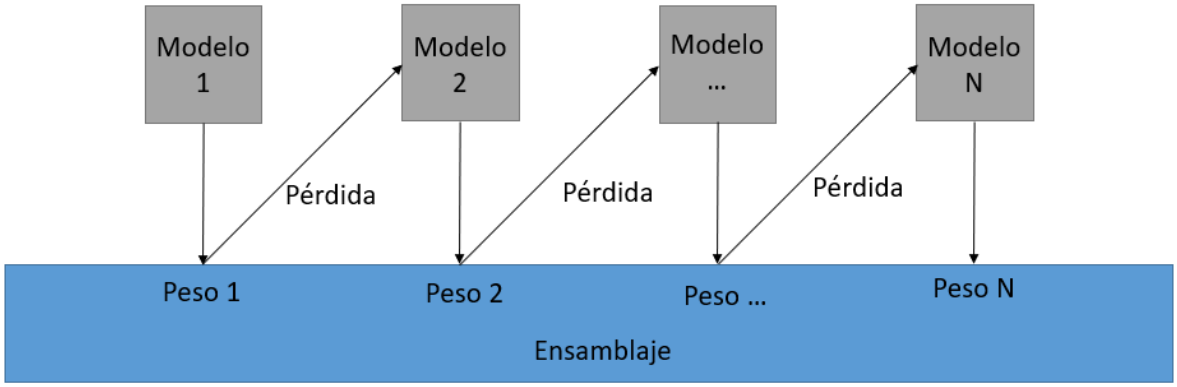


Ilustración 17. Ejemplo de funcionamiento del ensamblaje de modelos base del algoritmo AdaBoost. Fuente: Elaboración propia

A continuación, se va a explicar el funcionamiento matemático del algoritmo:

1. La ecuación (41) muestra el primer paso, que consiste en dar unos pesos al conjunto de datos, al ser la primera vez, todos los pesos son iguales.

$$\mathrm{w}(\mathrm{x_i}, \mathrm{y_i}) = \frac{1}{N}\ ,\ \mathrm{i} = 1,\ 2, \ldots,\ \mathrm{n} \tag{41}$$

   donde N es el número total de datos.
2. El siguiente paso es crear un tocón de decisión para cada una de las características del conjunto de datos y calcular el índice de Gini de cada árbol. El árbol que tenga el índice de Gini más bajo será el primer tocón.

3. A continuación, se calcula la importancia o influencia o Amount of Say (AoS) para este modelo utilizando la ecuación (42):

$$\mathrm{AoS}\ (\alpha) = \frac{1}{2} log_e \frac{(1 - ErrorTotal)}{ErrorTotal} \tag{42}$$

   Siendo el error total la suma de todos los pesos muestrales de los datos mal clasificados.
   El resultado puede dar valores entre 0 y 1, indicando los valores próximos a 0 que el tocón es bueno y los valores próximos a 1 valores indicando que el tocón es malo.

4. De esta forma, las predicciones erróneas tienen más peso que las correctas. Así al construir el siguiente modelo, se dará más importancia a estos puntos. La ecuación (43) contiene la fórmula para actualizar los pesos.

$$w_{\mathrm{nuevos}} = w_{\mathrm{antiguos}} e^{(\pm \propto)} \tag{43}$$

Una vez están actualizados los pesos, estos se normalizan.

5. A continuación, hay que hacer un nuevo conjunto de datos para ver si los errores han disminuido. Para ello, Dividimos nuestro conjunto de datos en paquetes, siendo el tamaño de estos los nuevos pesos normalizados.

6. Ahora, el algoritmo selecciona números aleatorios entre 0 y 1. Ya que los registros mal clasificados tienen pesos más altos, la probabilidad de seleccionarlos es más alta.

7. Por último, este nuevo conjunto de datos actuará ahora como nuestro conjunto de datos, por lo que, si se quieren obtener mejores resultados, habrá que repetir este proceso hasta obtener mejores resultados.

2.4.5.7. XGBoost

Existe una librería de código abierto que implementa algoritmos de Gradient Boosting optimizados distribuidos bajo el marco de trabajo de Gradient Boosting [79] [80]. Esta librería mezcla algunos conceptos que se han visto previamente, como árboles de decisión y Gradient Boosting.

XGBoost es un acrónimo de Extreme Gradient Boosting, esta biblioteca distribuida y escalable que implementa una serie de árboles de decisión de gradiente optimizado, o en inglés Gradient Boosting Decision Trees (GBDT).

GBDT es un algoritmo de aprendizaje conjunto parecido a Random Forest que mezcla árboles de decisión y Gradient Boosting para poder producir un mejor modelo.

Este algoritmo es muy similar a Random Forest ya que los dos construyen un modelo formado por varios árboles de decisión. La diferencia se da en el método de construcción y combinación de estos.

Mientras que Random Forest utiliza la técnica de *bagging* para construir los árboles en paralelo a partir de muestras Bootstrap y dando como predicción final una media de todas las predicciones de los árboles, GBDT mejora los modelos débiles combinándolos con otros modelos débiles para que el resultante sea fuerte en conjunto. Este proceso se realiza con un algoritmo de descenso de gradiente sobre una función objetivo. El Gradient Boosting establece los resultados objetivo para el siguiente modelo con el

objetivo de minimizar los errores. Los resultados se basan en el gradiente del error con respecto a la predicción.

Así, GBDT entrena de forma iterativa un conjunto de árboles de decisión poco profundos y, en cada iteración, utiliza los residuos del modelo anterior para ajustar el siguiente modelo. La predicción final es una suma ponderada de todas las predicciones de todos los árboles. El conjunto de árboles de decisión minimiza la varianza y reduce el riesgo de sobreajuste, mientras que el refuerzo de los árboles débiles con el Gradient Boosting minimiza el sesgo y reduce el riesgo de desajuste.

Con todo esto, se ha podido ver que XGBoost es una aplicación escalable y muy precisa que mejora el rendimiento de otros algoritmos como Random Forest ya que, en vez de construir los árboles secuencialmente, XGBoost sigue una estrategia por niveles evaluando la calidad de las divisiones en cada división posible del conjunto de entrenamiento.

Así, las ventajas de *XGBoost* son:

- Hay una gran cantidad de científicos de datos de todo el mundo que contribuyen de forma activa al desarrollo de código abierto de XGBoost.
- XGBoost funciona muy bien en muchos problemas distintos, pudiendo ser estos de regresión, clasificación, ranking, etc.
- XGBoost es una biblioteca disponible actualmente en los tres sistemas operativos más comunes: Linux, Windows y OSX.
- Permite integración en la nube gracias a AWS o Azure.
- Es una librería muy utilizada en múltiples organizaciones.
- XGBoost es una biblioteca creada desde cero para ser eficiente y flexible.

2.5. Flujo de trabajo aplicados para el entrenamiento de algoritmos de aprendizaje automático

Para el proceso de entrenamiento de algoritmos se utilizará un único conjunto de datos que contendrá únicamente cuatro atributos: latitud, longitud, edad, calidad de vida. A continuación, se definirá el flujo de trabajo que se utilizará para el entrenamiento de algoritmos de aprendizaje automático:

2.5.1. Preprocesamiento

La primera fase del proceso de aprendizaje automático es preparar los datos para que el algoritmo sea capaz de leerlos y utilizarlos para realizar el entrenamiento. Así, el modelo necesita una matriz X compuesta de las variables independientes y un vector y con una única variable dependiente de estas.

- Matriz X: contiene los datos de latitud, longitud y edad del habitante que vive en ese acceso.
- Vector y: contiene el valor de la calidad de vida que corresponde a esos datos.

2.5.1.1. División

Una vez se ha realizado la división de los datos, el conjunto de datos se divide en dos conjuntos diferentes, uno servirá como conjunto de entrenamiento y otro como conjunto de validación. La división se realizará guardando un 80% del conjunto de datos para el entrenamiento y el 20% restante servirá para validación, además se le aplicará el atributo *random_state* que modifica el orden del conjunto de datos para que no se deje ninguna zona sin entrenar. Esta división se hará con la función *train_test_split* de la librería *Scikit Learn*, en la siguiente línea se puede ver la sintaxis con la que se ha aplicado:

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=1)
```

2.5.1.2. Normalización

El siguiente paso es normalizar la matriz X. La ecuación (44) muestra como es el proceso de normalización:

$$z = \frac{x - \mu}{\sigma} \tag{44}$$

Donde μ es la media de los valores. La ecuación (45) explica cómo se calcula este valor:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{45}$$

Donde σ es la desviación estándar de los valores. La ecuación (46) explica cómo se calcula este valor:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad (46)$$

Este proceso es muy importante porque trabajar con datos a diferentes escalas afecta negativamente al modelo produciendo un resultado sesgado. Así que tras el proceso de normalización todos los datos se quedan en una escala de 0 a 1.

El proceso de normalización se realiza mediante la función *StandardScaler* de la librería *Scikit Learn*. Esta función permite normalizar los valores y para aplicarla es tan fácil como llamar a la función utilizando el siguiente código:

```
sc = StandardScaler()

X_train = sc.fit_transform(X_train)

X_test = sc.transform(X_test)
```

2.5.2. Importación de los algoritmos y entrenamiento

Una vez los datos ya están divididos y estandarizados es el momento de importar el algoritmo y entrenarlo. El proceso de importación consiste en escribir la función que se va a utilizar, siendo una función diferente para cada algoritmo, indicando los parámetros del algoritmo. El proceso de entrenamiento se realiza utilizando el código *regressor.fit(X_train, y_train)*. Este método permite entrenar el algoritmo guardado en la variable *regressor* con los datos de entrenamiento guardados en las variables *X_train* e *y_train.* Una vez terminado el entrenamiento el modelo está listo para predecir.

2.5.3. Cálculo de las métricas de rendimiento

Una vez se ha terminado el entrenamiento del modelo hay que comprobar cómo es el rendimiento del modelo que ha producido. Para comprobar esto se utilizan distintas métricas de rendimiento. Para obtener estas métricas de rendimiento, lo primero que se debe hacer es realizar una predicción utilizando el conjunto de datos de validación. Esto se hace con una sola línea de código: *y_pred = regressor.predict(X_test).* Esta línea guarda en la variable *y_pred* un vector y con el conjunto de datos predichos por el modelo. El siguiente paso es comparar la diferencia que hay entre la variable *y_pred* con *y_test*.

Las métricas de rendimiento que se van a calcular son la puntuación r^2, el error medio absoluto (MAE), el error medio cuadrático (MSE) y la raíz del error medio cuadrático

(RMSE). Estas métricas se calculan utilizando las funciones *r2_score, mean_absolute_error* y *mean_squared_error*, provenientes de la librería *Scikit Learn*.

Otra métrica de rendimiento que se utilizará es la validación cruzada. Esta métrica se puede utilizar gracias a la función *cross_val_score* de la librería *Scikit Learn* y permite obtener métricas como la media o la desviación estándar

2.5.4. Búsqueda de modelos óptimos

Tras calcular las métricas de rendimiento se puede dar el caso de que estas no sean muy buenas, en ese caso se probará otra vez el algoritmo variando los parámetros de este con el fin de obtener mejores resultados.

La función *GridSearchCV* de la librería *Scikit Learn* permite agilizar este proceso, ya que con introducir una serie de parámetros del algoritmo que se desea probar, la propia función prueba todas las combinaciones devolviendo los parámetros del algoritmo que pueden generar mejores resultados. Después con los parámetros devueltos por esta función se repetirá el proceso de los apartados 2.5.3 y 2.5.4.

2.5.5. Guardar el modelo.

Una vez se ha encontrado un modelo que sea capaz de realizar buenas predicciones, el siguiente paso es guardar el modelo. El modelo se almacena como un archivo pkl en una ruta del ordenador, se puede guardar con la función *dump* de la librería *pickle*. Se puede hacer esto utilizando las dos siguientes líneas de código:

```
with open('Modelos/Accesos/Sin_Distancias/XGBoost.pkl','wb') as f:
    pickle.dump(regressor,f)
```

Por último, una vez que ya se tiene el modelo guardado, este se podrá volver a cargar en un futuro guardándolo en una nueva variable, se puede hacer mediante la siguiente línea de código:

```
Regressor=
pickle.load(open('Modelos/Accesos/Sin_Distancias/XGBoost.pkl', 'rb'))
```

Una vez ya se tiene guardado un modelo óptimo, se pueden realizar predicciones utilizando otros datos. Estos datos también deben pasar por un proceso de normalización como el descrito en el apartado 2.5.1.2. Por último, para predecir con un nuevo conjunto de datos solamente hay que utilizar el atributo *predict* de la variable que contiene el modelo.

3. MATERIAL

3.1 Datos

Para este proyecto se ha contactado con un municipio que se ofrezca a ceder los datos. El municipio que ha cedido los datos es Castellar del Vallès, en la provincia de Barcelona. Los datos cedidos por el municipio son las capas en formato *geojson* de accesos y actividades económicas, además de una serie de Puntos De Interés (POIs), los cuales son: administración, cultura, educación, deporte, sanidad, servicios y transporte. Además, para obtener un resultado más real de este proyecto, se ha realizado una encuesta tomando una muestra de personas en diferentes tramos de edad, en la que se pregunta por la importancia de tener cerca cada uno de estos indicadores.

A. **Accesos:** La capa de habitantes contiene los atributos de id, número de acceso, referencia catastral y geometría en formato de punto cuyo sistema de referencia es WGS84. La Tabla 1 muestra la composición del conjunto de datos de accesos a viviendas. Los atributos de esta tabla son: num_acceso, indica el número asignado a ese acceso a vivienda, ref_cat, indica la referencia catastral de la vivienda a la que está asignado el acceso, geometry, indica la ubicación del acceso a la vivienda.

Tabla 1. Conjunto de datos de accesos

id	num_acceso	refcat	geometry
0	1	24	2483048DG2028S
1	2	5-11	3959016DG2035N
2	3	78	4377801DG2047N
3	36	4	4281511DG2048S
4	37	6	4281512DG2048S

- **POIs:** Son varios conjuntos de datos, estos son: Administración, Cultura, Educación, Deporte, Ocio, Sanidad, Servicios y Movilidad.

Todos los conjuntos de datos aquí tienen la misma estructura. Los atributos que contienen los atributos son f_id, poi_category, poy_type, poy_nom, poi_adreca, poi_telefon, poi_observacions, tipo_poi, categoría, idioma, atributo y geometry en formato de punto cuyo sistema de referencia es WGS84.

B. **Actividades económicas**: contiene los atributos de id, categoría, subcategoría, nombre, dirección, teléfono, horario, observaciones, idioma y geometría en formato de punto cuyo sistema de referencia es WGS84. La Tabla 2 muestra la composición del conjunto de datos de accesos a viviendas. Los atributos de esta tabla son: categoría, indica el tipo de actividad económica que realiza el comercio, aa_ida, indica el identificador del comercio, geometry, indica la ubicación del comercio.

Tabla 2. Conjunto de datos de actividades económicas

	Id	**categoría**	**aa_ida**	**geometry**
0	904	106###PEIXETERIA	ACTIVIDAD_012253	POINT (2.08810 41.61469)
1	1198	106###PEIXETERIA	ACTIVIDAD_012255	POINT (2.08754 41.61611)
2	914	106###PEIXETERIA	ACTIVIDAD_012256	POINT (2.08729 41.61523)
3	1223	110###PERRUQUERIES	ACTIVIDAD_013554	POINT (2.08657 41.61550)
4	1224	110###PERRUQUERIES	ACTIVIDAD_013556	POINT (2.08618 41.61327)

El atributo categoría indica la actividad económica que se realiza en ese lugar. Las actividades que aparecen en este conjunto de datos son: Peluquería., Alimentación., Ropa, Zapatería, Pescadería, Restaurante, Supermercado, Mecánico, Bar, Bazar, Bodega, Lavandería, Carnicería, Imagen y comunicación, Dietética, Droguería, Electricista, Electrodoméstico, Ferretería, Panadería, Frutería, Artículos de deporte y Muebles y decoración.

3.2 HARDWARE

Para la realización de este proyecto se utilizará un ordenador personal de sobremesa, que cuenta con un procesador Intel i7, una tarjeta gráfica Nvidia GTX 1050 y 16GB de memoria RAM.

3.3. SOFTWARE

Para este proyecto se utilizarán diferentes programas y a continuación se explicará en detalle cada software utilizado.

3.3.1. QGIS

QGIS [81], anteriormente llamado Quantum GIS, es una aplicación profesional de Sistemas de Información Geográfica de Software Libre y de Código Abierto (FOSS). En la Ilustración 18 se puede ver el logotipo de este software:

Ilustración 18. Logotipo de QGIS. Fuente: [82]

QGIS es un software muy útil ya que además de tener características propias de un software SIG, tiene algunas ventajas interesantes como poder trabajar con archivos ráster y vectoriales, también permite realizar una conexión directa con bases de datos (BBDD) como *PostgreSQL*, y con BBDD espaciales como PostGIS. Una de las mejores características que tiene QGIS, es que permite la instalación de complementos desarrollados en lenguajes de programación como *C++* o *Python* que permiten automatizar muchas tareas de este software. Además, actualmente funciona en cualquier sistema operativo (SO) que existe.

QGIS se utilizará para poder obtener una mejor visualización de los datos, también gracias a sus herramientas se realizará un mapa de calor que facilite la visualización de los resultados.

3.3.2. Python

Python [83] es un lenguaje de programación de alto nivel desarrollado por Python Software Fundation y posee una licencia de código abierto. En la Ilustración 19 se puede ver el logotipo de este software:

Ilustración 19. Logotipo de Python. Fuente: [84]

Python es actualmente uno de los lenguajes de programación más populares y actualmente se utiliza para desarrollar todo tipo de aplicaciones, algunas de ellas son Netflix o Instagram.

Python será el lenguaje de programación realizado para este proyecto, además del software principal de este, se ejecutará en un entorno *Jupyter-Notebook* de Anaconda [85] y a través de aquí, gracias al uso de diferentes librerías como NumPy, Pandas o Scikit Learn se realizarán todos los cálculos de este proyecto.

3.3.3. Jupyter-Notebook

Jupyter-Notebook [86] es una aplicación web de código abierto desarrollada por la comunidad de Proyecto Jupyter. En la Ilustración 20 se puede ver el logotipo de este software

Ilustración 20. Logotipo de Jupyter. Fuente: [87]

Esta aplicación permite compilar códigos de *Python* divididos en celdas y ejecutar cada una de estas de forma independiente, lo que resulta muy interesante para el análisis de datos y muchos procesos de cálculo.

3.3.4. NumPy

NumPy [88] es una librería del lenguaje de programación *Python* de código abierto que permite crear vectores y matrices multidimensionales, además esta librería también aporta un conjunto de funciones matemáticas de alto nivel. En la Ilustración 21 se puede ver el logotipo de esta librería:

Ilustración 21. Logotipo de NumPy. Fuente: [89]

3.3.5. Matplotlib

Matplotlib [90] es una librería del lenguaje de programación *Python* de código abierto que permite visualizar distintos gráficos y conjuntos datos. En la Ilustración 22 se puede ver el logotipo de esta librería:

Ilustración 22. Logotipo de Matplotlib. Fuente: [91]

Gracias a esta librería se podrán realizar las gráficas que se mostrarán a lo largo de este proyecto, además de que también permite crear y modificar imágenes.

3.3.6. Pandas

Pandas [92] es una librería del lenguaje de programación *Python* de código abierto que permite el manejo de datos y es utilizada para el análisis de estos. En la Ilustración 23 se puede ver el logotipo de esta librería:

Ilustración 23. Logotipo de Pandas. Fuente: [93]

Pandas permite crear *conjuntos de datos* que facilita mucho el trabajo con los conjuntos de datos, pudiendo manipularlos para poder obtener resultados.

3.3.7. Scikit Learn

Scikit Learn [94] es una librería del lenguaje de programación *Python* de código abierto construida a partir de *NumPy*, *Scipy* y *Matplotlib*. En la Ilustración 24 se puede ver el logotipo de esta librería:

Ilustración 24. Logotipo de Scikit Learn. Fuente: [95]

La librería *Scikit Learn* ofrece herramientas para el análisis descriptivo. Además, esta librería también proporciona una gran cantidad de algoritmos de ML que servirán de gran ayuda para la realización de este proyecto.

3.3.8. CartoCiudad

CartoCiudad [96], es un Software As A Service (SAAS), lo que significa que es un software online que se utiliza como servicio. CartoCiudad es un software público que utiliza el sistema cartográfico nacional para ofrecer su servicio. Algunos de los servicios que ofrece son direcciones postales, topónimos, poblaciones y límites administrativos de España; todos ellos se obtienen de fuentes oficiales.

Uno de los servicios que también ofrece CartoCiudad es calcular la distancia Manhattan entre dos puntos sin tener en cuenta el sentido de la circulación, por lo que es un servicio que interesa para este proyecto ya que el tramo se va a realizar andando. Se puede acceder a este servicio a partir del siguiente enlace: CartoCiudad. Para utilizar este servicio tan solo hay que indicar un punto de origen y uno de destino, y ya el propio servicio es el encargado de realizar los cálculos a través del callejero de la ciudad, tal y como se puede ver en la Ilustración 25 utilizando como ejemplo dos puntos de la ciudad de Madrid:

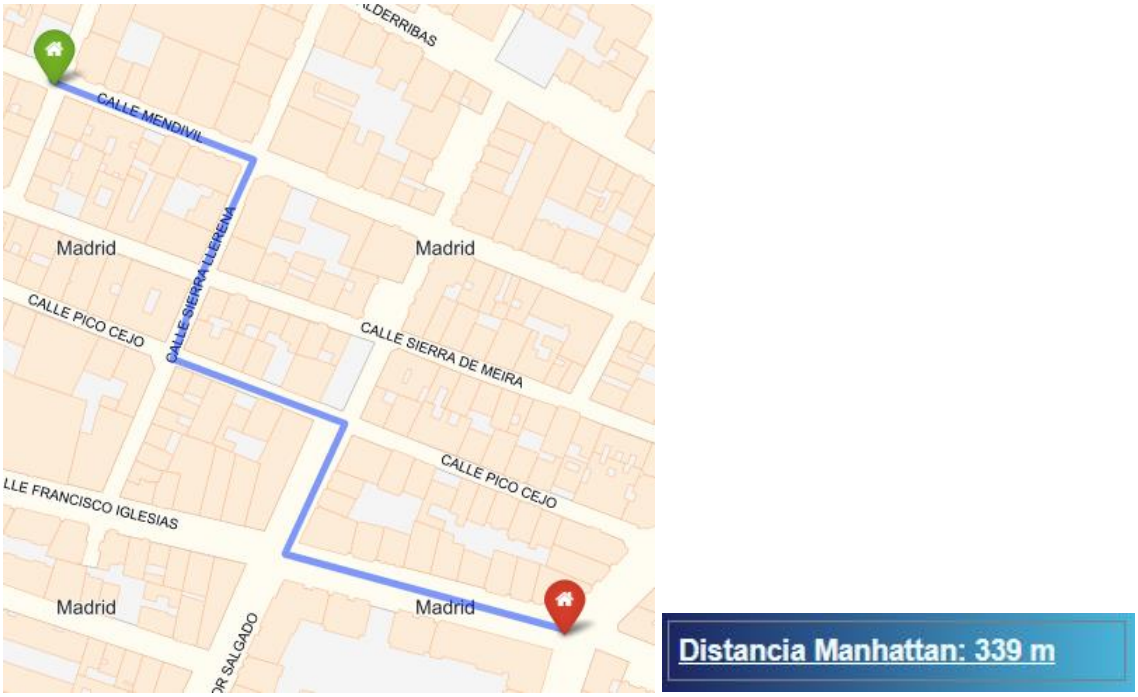


Ilustración 25. Ejemplo de uso del servicio de cálculo de distancias de CartoCiudad.
Fuente: [97]

Este servicio va a ayudar mucho a la realización del trabajo, pero existe el problema de que no se pueden pasar todos los puntos al servicio a la vez ya que este colapsaría, por lo que las peticiones hay que hacerlas una a una para que el servicio pueda responder a todas sin dar fallo, a este problema se le suma la cantidad de puntos que hay para cada vivienda, por lo que ejecutar el servicio para todos los puntos de los que se dispone no parece una opción que se pueda tomar. Para disminuir la cantidad de puntos con la que se está trabajando, se realizará un proceso previo que permitirá eliminar algunos puntos.

4. PROPUESTA E IMPLEMENTACIÓN DE LA METODOLOGÍA

En este apartado se explicará la metodología utilizada en este proyecto. La Ilustración 26 muestra un esquema con los apartados de este proyecto y con las horas que ha llevado la realización de cada uno:

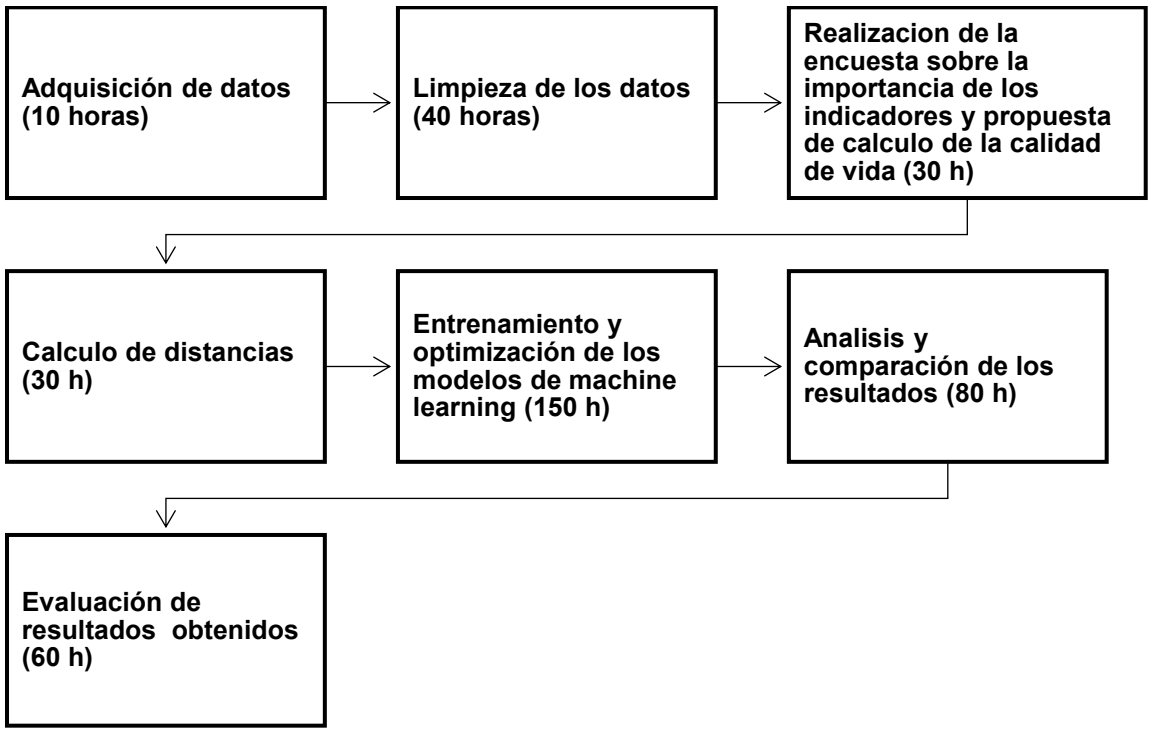


Ilustración 26. Resumen de la metodología aplicada en este proyecto. Fuente: Elaboración propia

4.1. Limpieza de datos y obtención de los indicadores

El primer paso una vez obtenidos todos los datos para la realización de este proyecto, es un proceso de limpieza de estos, este proceso también es conocido como *data cleaning*. Este proceso consiste en analizar todos los conjuntos de datos de los que se dispone, en el caso de este proyecto se dispone de tres conjuntos de datos distintos, los accesos a las viviendas, los POIs y las actividades económicas del municipio. Se analizará cada conjunto uno a uno para posteriormente, unificarlos en un único conjunto de datos.

Los tres conjuntos de datos tienen el atributo *geometry* en el mismo formato de geometría, además de que también está en el mismo sistema de referencia para los tres puntos. Por lo que el primer paso común para los tres conjuntos de datos es crear dos nuevos atributos a cada conjunto, uno para la latitud y otro para la longitud. Estos dos nuevos atributos llenarán sus campos con la información que hayan obtenido del atributo *geometry*. A partir de aquí, se realizará una limpieza de datos a cada conjunto individualmente.

4.1.1. Operaciones de limpieza de Accesos

La capa de accesos a viviendas se va a utilizar como punto de referencia para la calidad de vida, por lo que de esta capa solo nos interesan sus coordenadas, así que se eliminarán todos los atributos menos la latitud y la longitud. La Tabla 3 muestra cómo queda el conjunto de datos de accesos una vez terminado el proceso de limpieza.

Tabla 3. Conjunto de datos de accesos limpio

	Latitud	**Longitud**
0	41.621468	2.068474
1	41.600270	2.085002
2	41.616524	2.089927
3	41.618820	2.089480
4	41.618908	2.089475

4.1.2. Operaciones de limpieza de Actividades económicas

La capa de actividades económicas contiene todos los tipos de comercios en un mismo conjunto de datos. El primer paso para la limpieza de datos consistirá en crear un nuevo conjunto de datos para cada uno de los diferentes comercios que están contenidos en el conjunto de actividades económicas. Esto se puede hacer filtrando por el atributo *category* que contiene la categoría del comercio que está en esa ubicación.

Una vez se ha tiene un conjunto de datos para cada tipo de comercio diferente, se les aplicará a todos la misma limpieza, ya que al pertenecer todos al mismo conjunto de datos, tienen los mismos atributos.

De las actividades económicas del municipio se va a calcular su distancia a cada vivienda, por lo que lo único que interesa de este conjunto de datos es la ubicación de cada comercio y el tipo de comercio que es.

Como el tipo de comercio está declarado en cada conjunto de datos, solo interesan los atributos de latitud y longitud, pudiendo así eliminar todos los atributos menos estos dos.

Tras terminar este proceso, se obtienen varios conjuntos de datos, uno por cada tipo de actividad económica, que tienen la misma estructura que la tabla 1.

4.1.3. Operaciones de limpieza de POIs

Los POIs son varios conjuntos de datos, todos ellos con la misma estructura. Estos conjuntos de datos tienen el atributo *poi_type* que contiene distintos tipos de ubicaciones dentro de ellos; todos los tipos de ubicación contenidos en este atributo están indicados en el apartado 3.1 Datos. Así, se obtiene un conjunto de datos distinto para cada tipo de POI.

Al igual que con los conjuntos de datos de actividades económicas, se va a calcular la distancia de cada acceso a la vivienda a todos los POIs, por lo que la latitud y la longitud son los dos únicos atributos que nos interesan de estos conjuntos de datos, pudiendo así eliminar todos los atributos restantes y obteniendo un resultado final para cada conjunto como el mostrado en la tabla 1.

4.2. Cálculo de distancias y tiempo en recorrerla en 15 minutos

Una vez terminado el proceso de limpieza para todos los conjuntos de datos, el siguiente paso es saber la distancia que hay desde cada punto de acceso a cada POI o actividad económica, la distancia que se va a utilizar para el proyecto es la distancia Manhattan, ya que esta distancia sigue el recorrido de la calle y es más útil para saber el tiempo que tarda una persona en ir a ese punto andando.

Para calcular esta distancia se va a utilizar el servicio de ruta de CartoCiudad, el cual se explica en el apartado 3.3.8. Pero, existen una serie de problemas que hay que resolver previamente.

El problema consiste en que al tratarse de un servicio web, cada punto que se desea calcular es una petición al servidor, y para este proyecto se deben calcular 2.826.284 distancias, lo cual es un número que el servicio no se puede permitir, ya que de realizar todas estas peticiones de golpe el servicio colapsaría. Sabiendo esto, se realizarán todas las peticiones una a una para que el servicio sea capaz de responder a todas sin dar fallo.

Al tener que responder todas las peticiones una a una, nace un nuevo problema, y este vuelve a estar relacionado con la cantidad de peticiones que se deben hacer. Por lo que será necesario realizar una reducción de las distancias a calcular.

La solución que se propone para resolver este problema es calcular solo la distancia de las ubicaciones que estén a menos de 15 minutos, ya que para este proyecto solo se tendrán en cuenta las ubicaciones que estén a menos de 15 minutos caminando. Por lo que una opción para reducir la cantidad de puntos a calcular es calcular primero la distancia euclídea a todos los puntos y quedarnos únicamente con los que tengan una distancia menor de 1500 metros, se escoge esta distancia ya que es la distancia que tarda 15 minutos en recorrer la persona más lenta que se tiene en cuenta para este proyecto, según un estudio que mide las velocidades caminando de las personas, el cual se puede ver en la web de *causadirecta* [98] y se profundizará en el más adelante. Se tiene en cuenta la distancia euclídea porque es la distancia más corta entre dos puntos, por lo que, si esta es superior a 15 minutos, cualquier otra distancia también lo va a ser.

Gracias a esto se eliminan una gran cantidad de números, pasando de tener que calcular 2.826.284 a tan solo 502.799. Esto supone una reducción considerable, ya que tiene que realizar el 17,79% del total de cálculos. Tras terminar todo este proceso de cálculo, se obtiene un único conjunto de datos como el que se puede ver en la Tabla 4. Este conjunto de datos contiene la distancia mínima de cada ubicación a cada tipo de POI o actividad económica más cercana, estando estas nombradas por números para facilitar el cálculo. Este conjunto de datos tiene un total de 41 atributos, uno para el índice, dos para las coordenadas, y 38 para las distancias a los POIs y actividades económicas.

Tabla 4. Conjunto de datos con las distancias calculadas

	Latitud	**Longitud**	**0**	**1**
0	41.621468	2.068474	1679.174531	1693.280484
1	41.600270	2.085002	252.006124	1623.817564
2	41.616524	2.089927	210.745890	203.514087
3	41.618820	2.089480	334.710363	341.637155
4	41.618908	2.089475	342.875009	350.029558

Como última parte del cálculo de distancias, en el proceso de análisis se pide que tenga una ubicación en el rango de 15 minutos, por lo que se realizará una comparación de la

distancia entre cada tipo de punto y se seleccionará únicamente la menor de todas, ya que esta será la ubicación a la que vaya el habitante de ese acceso.

Una vez ya se tiene un conjunto de datos con la ubicación de cada acceso y la distancia a cada POI o comercio más cercano se puede calcular el tiempo que se tarda en recorrer esa distancia. Esto se puede calcular gracias a la ecuación de la velocidad, despejando el tiempo de esta ecuación, tal y como se indica en la ecuación (47):

$$v = \frac{d}{t} \rightarrow t = \frac{d}{v} \tag{47}$$

Donde:

- t: Tiempo que tarda en recorrer la distancia. Se mide en segundos
- d: Distancia que recorre. Se mide en metros
- v: Velocidad a la que recorre la distancia. Se mide en metros por segundo.

En el paso anterior se ha calculado la distancia, pero falta la velocidad a la que se recorre esa distancia para poder obtener el tiempo. Este dato se obtendrá del siguiente estudio realizado por causa directa, del cual se obtienen las conclusiones que se pueden ver en la *Tabla 5*:

Tabla 5. Velocidad a la que camina una persona según su edad. Fuente: *[98]*

EDAD (AÑOS)	MUESTRA (Nº PERSONAS)	VELOCIDAD (M/S)
5 – 9	26	1,8
10 – 14	37	1,65
15 – 19	47	1,62
20 – 24	65	1,59
24 – 34	70	1,59
35 – 44	67	1,59
45 - 44	73	1,5
55 - 64	90	1,44
65+	67	1,26

Como se puede ver en la tabla anterior, las velocidades varían en función de la velocidad, por lo que la distancia que se recorre en 15 minutos también varía según la edad. Para poder tener en cuenta la velocidad de cada persona, se construirá un nuevo conjunto de datos por cada rango de edad presente en este estudio, para calcular el tiempo que tarda una persona media de ese rango de edad en recorrer la distancia, de

esta forma se obtendrán nuevos conjuntos datos distintos correspondientes a cada rango de edad. Cada nuevo conjunto tendrá la misma estructura que el presentado en la Tabla 4.

4.3. Encuesta sobre la importancia de cada indicador y propuesta de la fórmula de cálculo de la calidad de vida

Una vez ya se han obtenido los datos, ya se puede tener una idea más clara de cómo dividir el proyecto. La idea es dividir todo este conjunto de datos en una serie de indicadores que son los que se tendrá en cuenta para calcular la calidad de vida de cada acceso. Este cálculo se puede hacer de varias formas, la forma más sencilla es calcular todos los indicadores y hacer un promedio de todos para sacar el valor de calidad de vida, pero lo cierto es que para las personas no todos los indicadores son igual de importantes.

Una opción para obtener unos valores más reales de la calidad de vida es establecer unos pesos a cada indicador según la importancia que estos tengan, pero no se pueden dar unos valores genéricos a la importancia de cada indicador, ya que a ciertos indicadores no le darán la misma importancia una persona de 15 años que una de 70.

Teniendo en cuenta todos estos factores, se hará una división por rangos de edad. Teniendo así que calcular la calidad de vida para un acceso según la importancia que le da a los indicadores cada rango de edad.

Con el fin de que los resultados se ajusten lo más posible a la realidad, se realizará una encuesta para saber la opinión de las personas sobre la importancia que se le da a cada indicador según el rango de edad.

La encuesta que se ha realizado consiste en una serie de preguntas, siendo la primera pregunta la edad de la persona que contesta, para darle más valor a su respuesta según el rango de edad, y después sigue la importancia que se le da a cada uno de los 12 indicadores que se tienen en cuenta para este proyecto, siendo las respuestas posibles un valor del 0 al 4 indicando 0 que no tiene nada de importancia y 4 que tiene mucha importancia.

4.3.1. Agrupación de puntos de interés en indicadores a utilizar en la predicción de la calidad de vida

Para facilitar el cálculo de la calidad de vida en cada ubicación, se han agrupado todos los POIs y comercios en una serie de indicadores, teniendo esta agrupación en cuenta para el cálculo. El criterio para la división de los indicadores se basará en los indicadores nombrados por el INE que se pueden ver en el apartado **¡Error! No se encuentra el origen de la referencia.** realizando algunas modificaciones que se consideran necesarias, como la división de consumo en comercios principales y secundarios, o sanidad en centros de salud o farmacias para comprar medicamentos. Así, se obtienen los siguientes indicadores a partir de los datos proporcionados:

1. **Área de descanso**. Esta categoría contiene los puntos de áreas de descanso y parques.
2. **Espacio cultural**. Este indicador incluye todas las ubicaciones que se relacionen con la cultura. Estas son bibliotecas, exposiciones y espacios culturales.
3. **Centro educativo**. Este indicador incluye todos los centros donde se imparta educación. Estos son guarderías, colegios, institutos y otros centros.
4. **Farmacias**
5. **Centros de salud**. Este indicador incluye todos centros de atención primaria y de cruz roja.
6. **Residencias**
7. **Centros deportivos**
8. **Servicios públicos**. Este indicador incluye ubicaciones que ofrezcan servicios públicos como el ayuntamiento, correos o notarías.
9. **Lugares de ocio**. Este indicador contiene los puntos que se consideran de ocio. Estos son bares, restaurantes, actividades recreativas y lugares de interés.
10. **Estaciones de transporte público**. Este indicador incluye estaciones de bus, taxi y transporte a demanda.
11. **Comercios básicos**. Este indicador contiene todos los comercios que se consideran de primera necesidad. Estos son los supermercados, pescaderías, bazares, panaderías, carnicerías, tiendas de alimentación no especializadas, mercados municipales y mercados ambulantes.
12. **Comercios secundarios**: Este indicador contiene todos los comercios que no son de primera necesidad para la mayoría de la población. Estos son electricistas, fontaneros, ferreterías, floristerías tiendas de electrodomésticos, muebles y ropa.

4.3.2. Espacio muestral

A esta encuesta han respondido un total de 41 personas, lo cual es un número de personas válido para poder tener en cuenta los resultados de la encuesta. A continuación, se realizará un análisis de los resultados para tener una idea de la opinión de la gente sobre cada indicador:

Lo primero para el análisis es ver la gente que ha respondido a la encuesta, la Ilustración 28 indica el número de personas que ha votado según la edad que tienen:

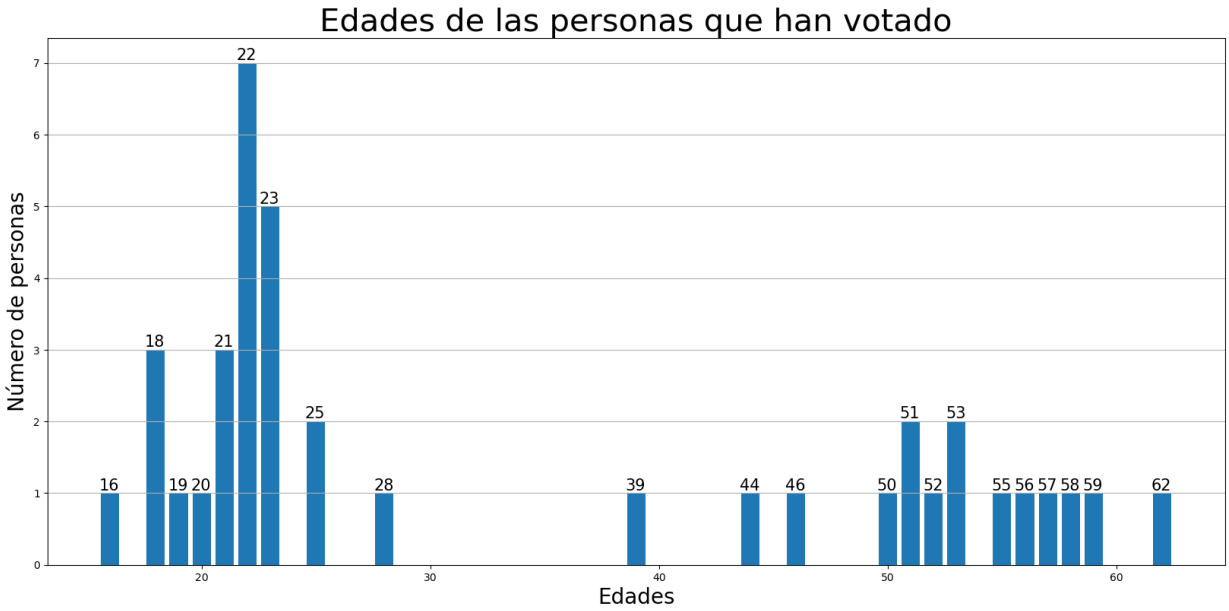


Ilustración 27. Gráfico de barras de la edad de personas que han votado en la encuesta.

Fuente: Elaboración propia

En este gráfico se pueden ver varios detalles, la edad de las personas que más han votado en la encuesta es 22 años, siendo este valor un 17% del total de votos, además, el 46% de las personas que han votado se encuentran entre 20 y 30 años. También se puede ver que la media de edades de voto es de 33 años con una desviación estándar de 33 años. También hay que destacar de no ha votado ninguna persona del tercer rango de edad y del primer rango de edad solo una persona.

Esta población muestral se ha divido en tres rangos codificando los rangos de edades correspondientes a los menores de edad (grupo 1, agrupando los respondientes con edades entre 0 y 18 años), a los mayores de edad entre 18 y 65 años (grupo 2), y a los mayores de edad con una edad superior a 65 años.

4.3.3. Análisis exploratorio de las respuestas

A continuación, se mostrarán las gráficas con las respuestas de cada indicador según el rango de edad para poder visualizar los resultados, después se hará un breve análisis de estos para intentar sacar algunas conclusiones que puedan ser interesantes.

Así, los resultados de los indicadores son los siguientes. La Ilustración 28 muestra las respuestas a los indicadores de áreas de descanso, espacios culturales, centros educativos y farmacias. La Ilustración 29 muestra las respuestas a los indicadores de centros de salud, residencias, centros deportivos y servicios públicos. La Ilustración 30 muestra las respuestas a los indicadores de lugares de ocio, estaciones de transporte público, comercios básicos y secundarios.

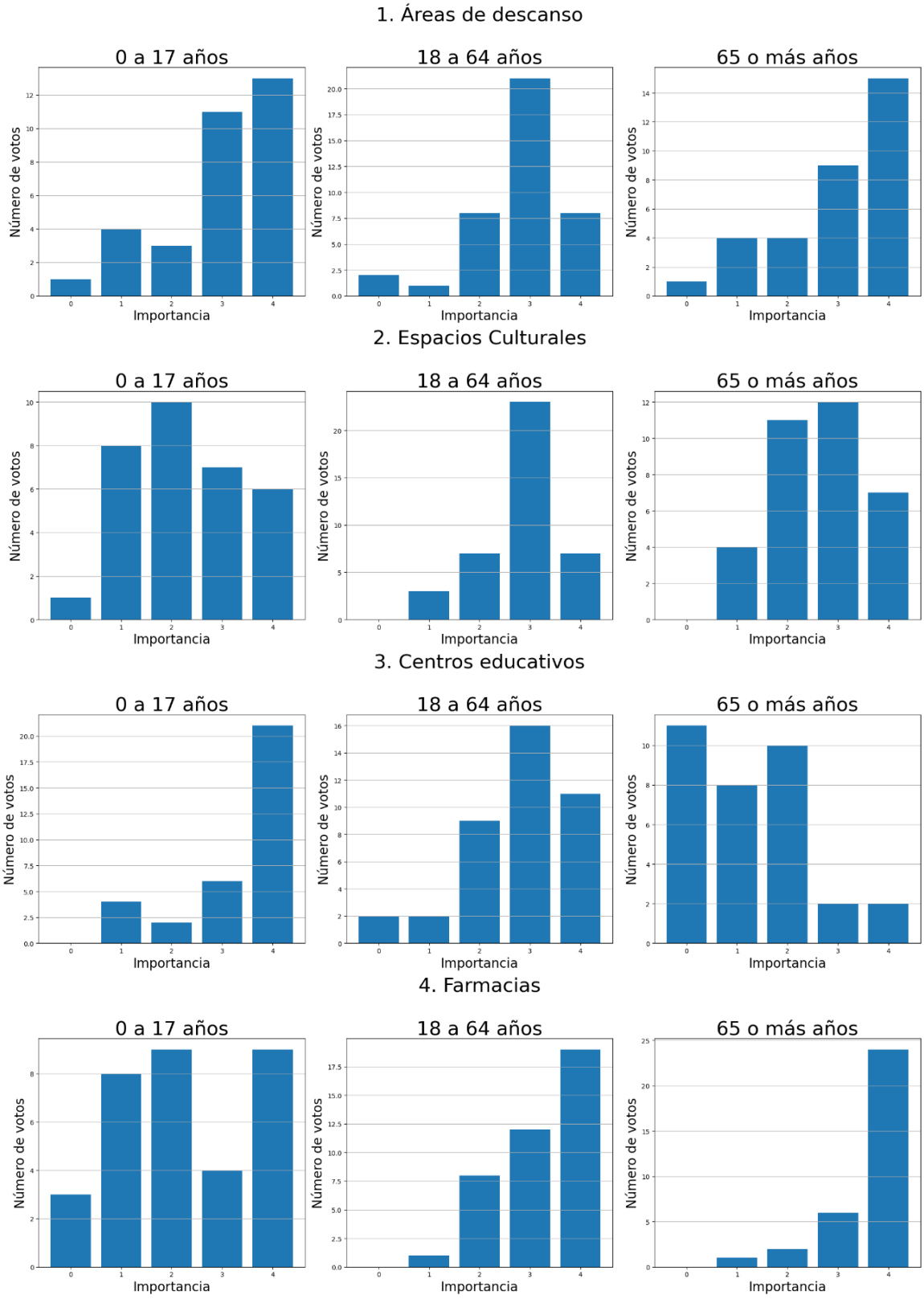


Ilustración 28. Histogramas con la importancia de los indicadores "Áreas de descanso", "Espacios Culturales", "Centro educativos" y "Farmacias" agrupadas por los rangos de edades considerados. Fuente: Elaboración propia

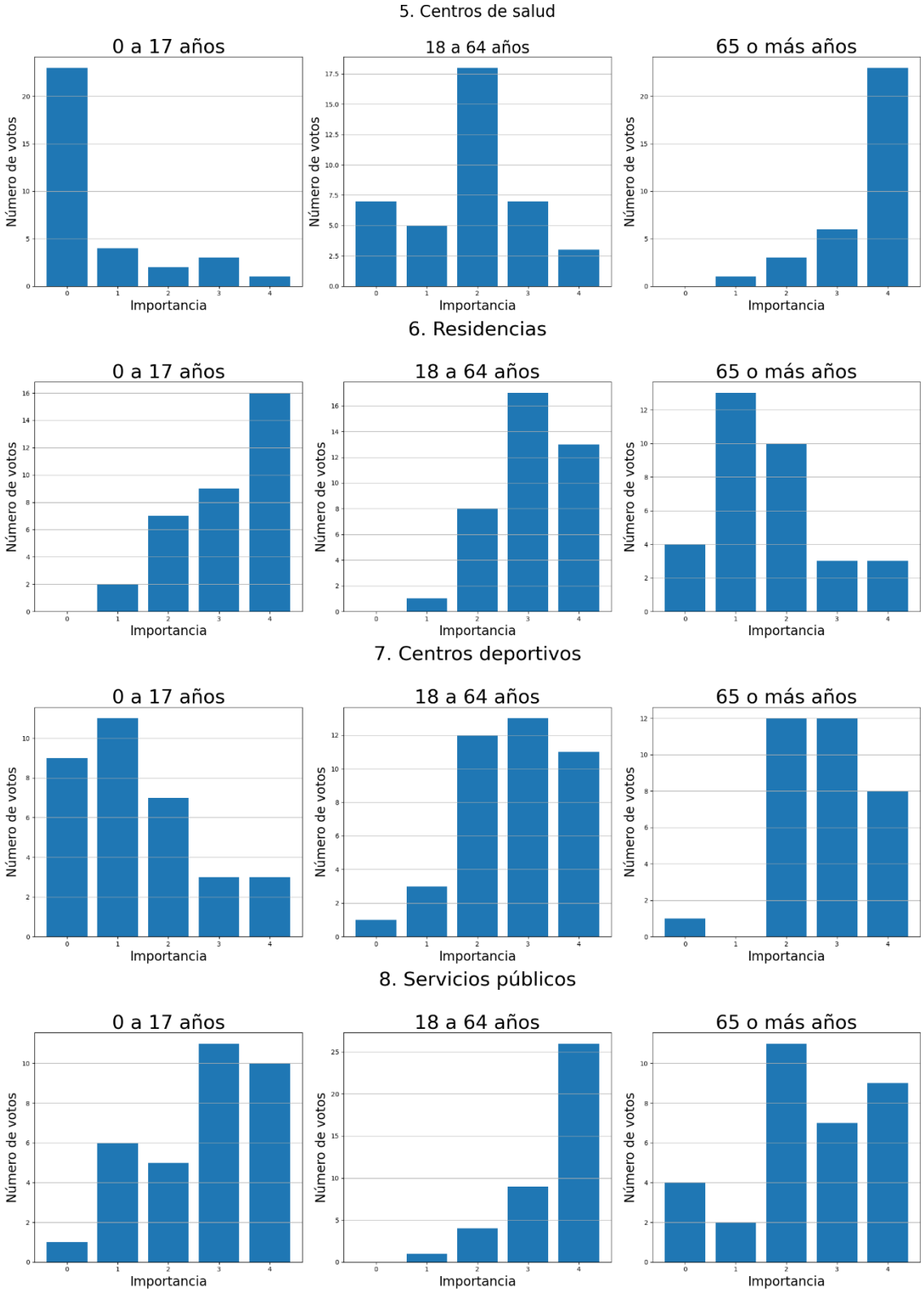


Ilustración 29. Histogramas con la importancia de los indicadores "Centros de salud", "Residencias", "Centro deportivos" y "Servicios públicos" agrupados por los rangos de edades considerados. Fuente: Elaboración propia

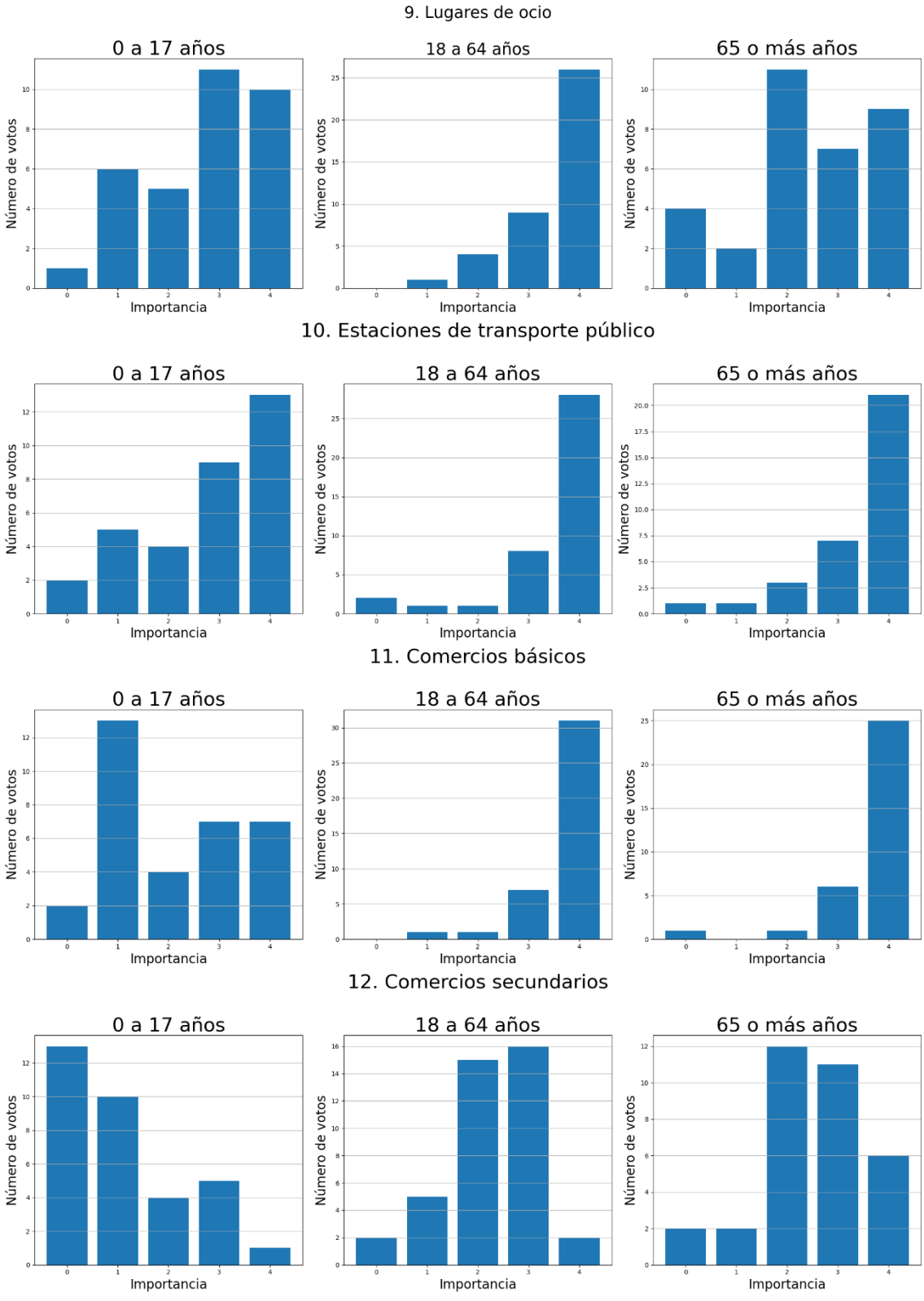


Ilustración 30. Histogramas con la importancia de los indicadores "Lugares de ocio", "Estaciones de transporte público", "Centro básicos" y "Comercios secundarios" agrupadas por los rangos de edades considerados. Fuente: Elaboración propia

Se pueden comprobar que en los resultados obtenidos apenas hay ningún tipo de variación según los rangos de edad. Se puede comprobar el resultado de una forma más numérica a partir de la Tabla 6 que muestra las medias y desviaciones de los resultados de cada pregunta:

Tabla 6. Media y desviación típica de cada indicador para cada grupo de edad

ID	Indicador	Grupos de edad					
		0 - 18 años		18 - 64 años		65+ años	
		Media	Desv.	Media	Desv.	Media	Desv.
1	**Áreas de descanso**	2,97	1,15	2,80	0,97	3,00	1,17
2	**Espacios culturales**	2,28	1,14	2,85	0,80	2,65	0,95
3	**Centros educativos**	3,33	1,05	2,80	1,07	1,27	1,18
4	**Farmacias**	2,24	1,35	3,23	0,86	3,61	0,75
5	**Centros de salud**	2,67	1,19	3,20	0,91	3,70	0,59
6	**Residencias**	0,64	1,14	1,85	1,14	3,55	0,79
7	**Centros deportivos**	3,15	0,96	3,08	0,81	1,64	1,11
8	**Servicios públicos**	1,39	1,25	2,75	1,03	2,79	0,93
9	**Lugares de ocio**	2,70	1,19	3,50	0,78	2,45	1,30
10	**Estaciones de transporte**	2,79	1,29	3,48	1,04	3,39	1,00
11	**Comercios básicos**	2,12	1,32	3,70	0,65	3,64	0,82
12	**Comercios secundarios**	1,12	1,19	2,28	0,93	2,52	1,06

De esta tabla se puede comprobar que no suele haber mucha duda sobre la importancia de cada indicador, ya que las desviaciones apenas superan una unidad.

Si se comprueba las variaciones de rango con el promedio total, también se puede ver que salvo en algunos casos siempre se muestran los mismos resultados. Esto se puede comprobar en la Tabla 7.

Tabla 7. Diferencia de cada rango de edad con el promedio

Indicadores	Promedio	0 - 17 años	18 - 64 años	65+ años
Áreas de descanso	2,92	0,05	-0,12	0,08
Espacios culturales	2,59	-0,31	0,26	0,05
Centros educativos	2,47	0,86	0,33	-1,20
Farmacias	3,02	-0,78	0,20	0,58
Centros de salud	3,19	-0,52	0,01	0,51
Residencias	2,01	-1,37	-0,16	1,53
Centros deportivos	2,62	0,53	0,46	-0,98
Servicios públicos	2,31	-0,92	0,44	0,48
Lugares de ocio	2,88	-0,19	0,62	-0,43
Estaciones de transporte	3,22	-0,43	0,26	0,18
Comercios básicos	3,15	-1,03	0,55	0,48
Comercios secundarios	1,97	-0,85	0,30	0,54

Se puede comprobar que todas las variaciones son próximas a cero, exceptuando en algunos indicadores como los centros educativos o residencias, donde llegan a variar más de un punto. También hay que destacar que el rango de menores de edad es el rango donde más varían los resultados.

Una vez realizado el análisis, es posible establecer los pesos en base a la importancia de cada indicador. Para establecer los pesos se ha tenido en cuenta el valor promedio de las respuestas según el rango de edad; además, estos valores se han normalizado para que el valor máximo de la calidad de vida sea para todos los rangos un 100%

Los pesos que se aplican a cada rango se pueden consultar en la Tabla 8 que muestra cada peso en forma de porcentaje:

Tabla 8. Matriz de pesos

Indicador	**0 – 17 años**	**18 – 64 años**	**65 o más años**
Áreas de descanso	10,72	7,80	8,69
Espacios culturales	8,36	8,02	7,59
Centros educativos	12,25	7,80	3,75
Farmacias	10,72	7,80	8,69
Centros de salud	9,66	9,14	10,80
Residencias	2,47	5,20	10,34
Centros deportivos	11,90	8,62	4,76
Servicios públicos	5,06	7,80	8,14
Lugares de ocio	9,78	9,88	7,32
Estaciones de transporte	10,01	9,73	10,06
Comercios básicos	7,66	10,40	10,61
Comercios secundarios	4,24	6,46	7,41

Con estos pesos establecidos, ya se puede construir una ecuación que permita calcular la calidad de vida de una ubicación siendo esta la ecuación (48):

$$qol = \sum_{i=1}^{n} v_i \mathrm{p}_i \tag{48}$$

Donde:

- v: Indica si el indicador se encuentra o no a menos de 15 minutos. Este valor es 1 o 0.
- p: Peso que se le da a ese indicador. El valor es obtenido de la tabla que contiene los pesos sacados de la encuesta.

4.4. Cálculo de Calidad de Vida basada en la distancia a cada indicador y preparación de formato de aprendizaje automático

Para calcular la calidad de vida descrita en la ecuación 47 definida en el apartado 4.3, se necesita saber si el indicador está a menos de 15 minutos de la vivienda, para después multiplicar este valor por el peso. Además, en el apartado 4.3 también se ha llegado a la conclusión de que es interesante dividir según rangos de edad, por lo que al final de este proceso se obtendrá un conjunto de datos distinto por cada rango de edad según la velocidad a la que camina una persona, pero los pesos que se aplicarán

a cada conjunto de edad variarán según el rango de edad que se ha tenido en cuenta para este proyecto.

Cada nuevo conjunto de datos tendrá los atributos de la latitud y la longitud del acceso a la vivienda, la edad de la persona que vive en ella, ya que como se ha visto antes la velocidad de una persona varía según su edad, aunque estén dentro del mismo rango de edad; y cada indicador que agrupa todos los puntos que se han tenido en cuenta para este proyecto, teniendo estos el valor de 1 si está a menos de 15 minutos o 0 si está a más. Esto se hace así porque facilita mucho el cálculo ya que solo hay que tener en cuenta si está dentro o no, y no dar un valor según la distancia que tenga.

Una vez están preparados los tres conjuntos de datos, se calcula la calidad de vida utilizando la ecuación 31 y esta se añade a cada conjunto de datos.

El último paso es unificar y preparar todos los conjuntos de datos para que los algoritmos de ML sean capaces de leerlos. Para esto se realizará una unión de todos los datos según la ubicación del acceso a la vivienda. El siguiente paso es eliminar todos los atributos que no interesan para el entrenamiento, estos son todos los indicadores.

Tras terminar todo este proceso se obtiene un único conjunto de datos que contiene la latitud y la longitud del acceso a la vivienda, la edad de la persona que vive en esa vivienda y la calidad de vida de esa persona. Tal y como se muestra en la Tabla 9.

Tabla 9. Conjunto de datos que se utilizará para el aprendizaje automático

	latitud	longitud	edad	calidad_vida
0	41.621468	2.068474	11	52.885748
1	41.600270	2.085002	12	74.793875
2	41.616524	2.089927	12	79.858657
3	41.618820	2.089480	11	79.858657
4	41.618908	2.089475	12	79.858657

4.5. Entrenamiento de modelos de Machine Learning

Para el entrenamiento de modelos de ML, se han entrenado los algoritmos explicados en el apartado 2.4.5, utilizando la librería Scikit Learn explicada en el apartado 3.3.7. Además, también se ha seguido el flujo de trabajo definido en el apartado 2.5.

4.5.1. Regresión Lineal

Este es el algoritmo más sencillo de todos, ya que de aquí se obtiene una simple recta, lo que hace muy probable que no pueda ajustarse correctamente al modelo.

Antes de aplicar el algoritmo, se realizará un análisis previo de los datos, con el fin de comprobar si es óptimo aplicar este algoritmo.

Lo primero es ver si existe algún tipo de correlación entre los datos, para ello se analizará la Ilustración 31 que muestra la correlación que hay entre cada atributo del conjunto de datos:

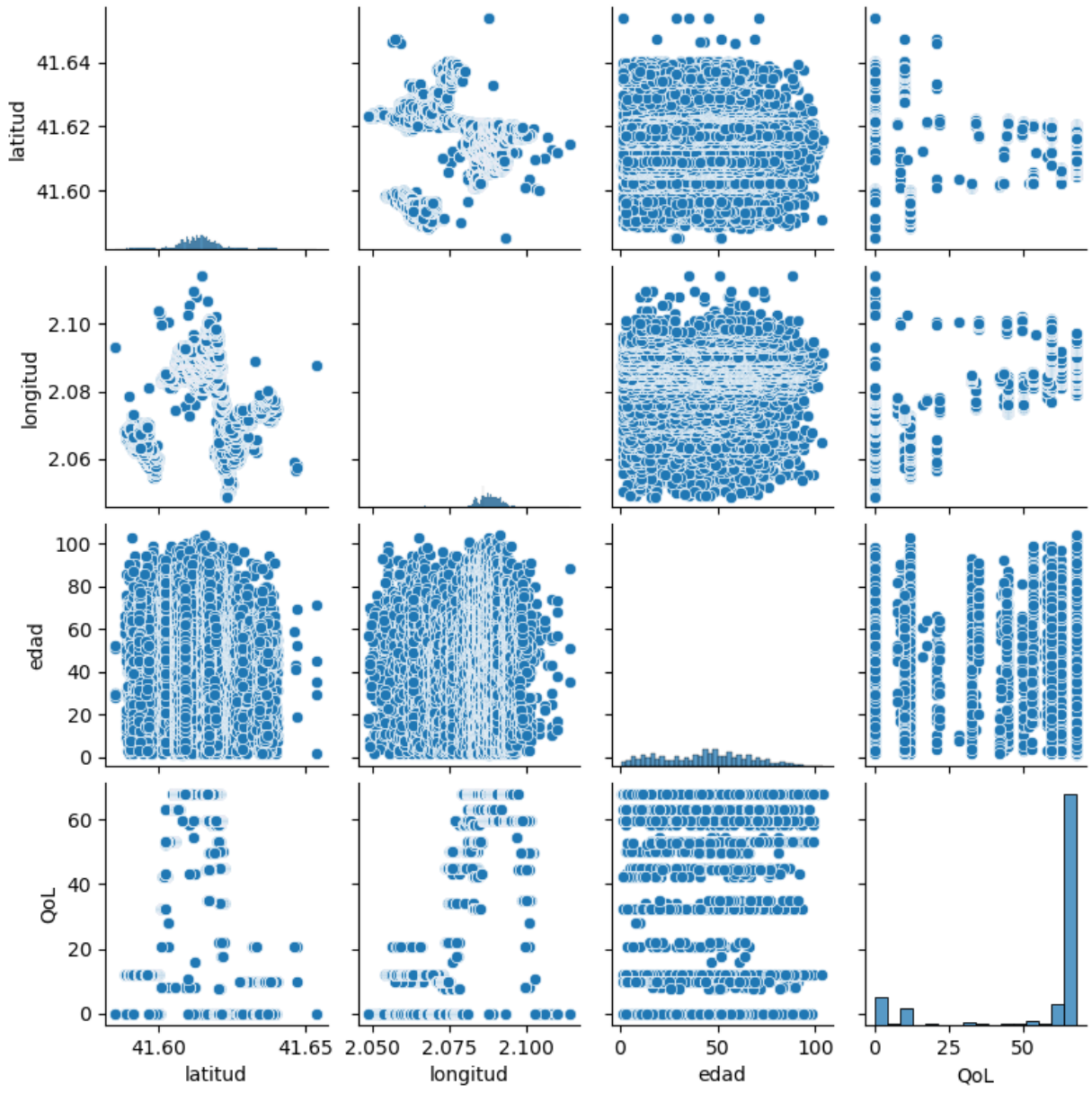


Ilustración 31. Correlación entre los datos. Fuente: Elaboración propia

No parece haber mucha correlación entre los datos, se mirará el coeficiente de correlación de Pearson con el fin de tener mayor seguridad al trabajar con un valor numérico. La Tabla 10 muestra el resultado de este cálculo:

Tabla 10. Coeficiente de correlación de Pearson

	Latitud	Longitud	Edad
Latitud	1	-0.029	-0.008
Longitud	-0.029	1	0.003
Edad	-0.008	0.003	1

También se puede mostrar el valor en forma de gráfico, tal y como muestra la Ilustración 32. Matriz de coeficiente de correlación de Pearson:

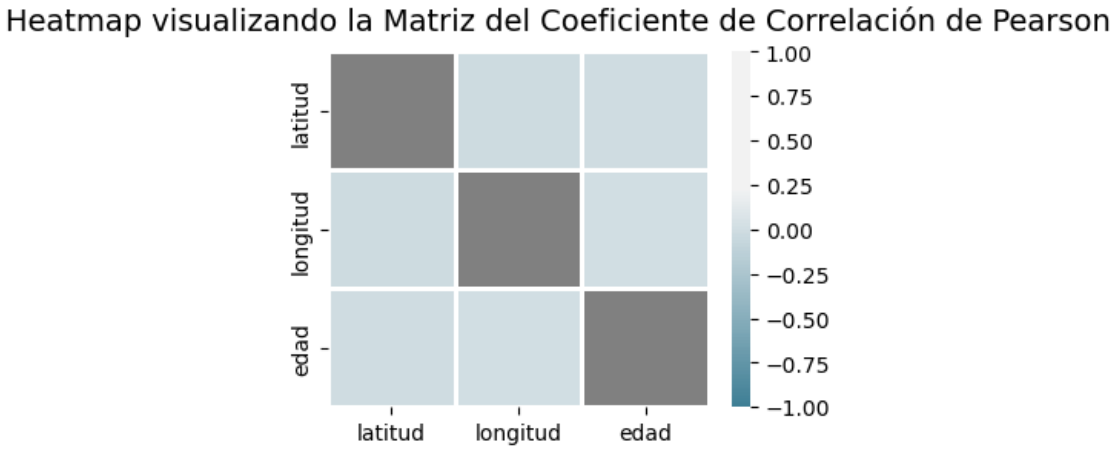


Ilustración 32. Matriz de coeficiente de correlación de Pearson. Fuente: Elaboración propia

Tras haber calculado el coeficiente de correlación de Pearson, este es prácticamente 0 en todos sus datos, por lo que se puede afirmar que no existe mucha correlación entre los datos.

Aun así, en ámbito de investigación, se va a utilizar el algoritmo para probar que tal se ejecuta el modelo generado.

La regresión lineal es un algoritmo muy simple por lo que no tiene hiper parámetros interesantes que modificar. El único que merece la pena probar es *intercept*, que en este caso ha sido positivo para que se añada a la ecuación.

Después de entrenar el modelo se han obtenido los siguientes coeficientes:

- β_0 = 67,11
- β_1 = -4.72

- β_2 = 15.1
- β_3 = -1.04

También se puede escribir los resultados de la regresión en forma de ecuación, obteniendo así la ecuación 48:

$$y = 4{,}72 \cdot lat + 15{,}1 \cdot long - 1{,}04 \cdot edad + 67{,}11 \quad (48)$$

Para este modelo se han obtenido un valor R2 de 0,72, un error medio absoluto de 8,36

- Error Medio Cuadrático = 136,07
- Raíz del Error Medio Cuadrático = 11,66

Son valores muy grandes por lo que conviene probar otros modelos mejores.

También se han comparado valores reales calculados mediante la ecuación 47 descrita en el apartado 4.3.3 con valores predichos por el modelo de regresión lineal que se acaba de calcular para ver las diferencias que hay. Esta comparación se puede comprobar en la Tabla 11:

Tabla 11. Valor calculado frente a valor predicho por el modelo de regresión lineal.

Valor calculado	**Valor predicho**
51,7	67,96
68,93	67,96
63,05	53,24
12,4	0
-3,17	0
59,41	67,96

Se puede ver que existen diferencias significativas entre los valores calculados con la ecuación 47 y los valores predichos por el modelo de regresión lineal.

A continuación, la Ilustración 33 muestra el hiperplano generado por el algoritmo, al ser un plano de más de 3 dimensiones, no se puede representar con una sola imagen, por lo que se muestran distintas imágenes en las que se relacionen las distintas variables:

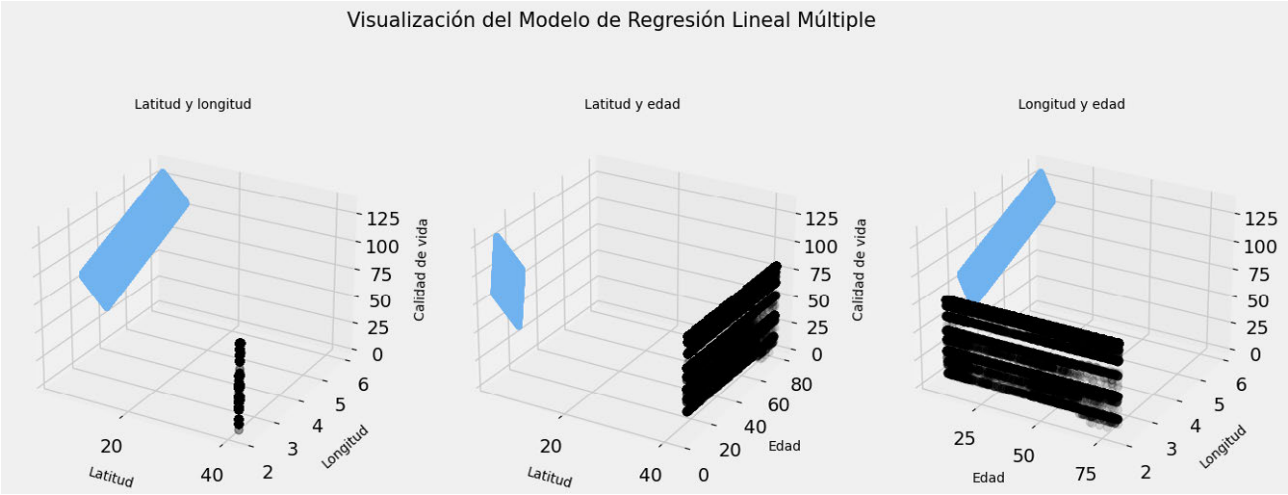


Ilustración 33. Hiperplano generado por el algoritmo de regresión lineal. Fuente: Elaboración propia

4.5.2. Support Vector Regressor

Otro algoritmo que se ha probado es SVR, como se ha visto antes, este algoritmo trabaja en base a un *kernel*, este *kernel* se puede presentar de cuatro formas distintas: lineal, polinómico, sigmoide o como función de base radial. Se ha probado el algoritmo con los cuatro tipos distintos de *kernel* para comprobar cual presenta mejores resultados. Los hiper parámetros utilizados para cada uno son los siguientes:

- RBF: C=100, Gamma=0.1, Epsilon=0.1
- Lineal: C=100, gamma="auto"
- Polinómico: C=100, gamma="auto", degree=3, epsilon=0.1, coef0=1
- Sigmoide: C=100, gamma="auto”

Así, se han entrenado cuatro modelos distintos, los resultados obtenidos para cada modelo son los que se muestran en la Tabla 12:

Tabla 12. Métricas de cada kernel de SVR

MÉTRICAS	RBF	LINEAL	POLINÓMICO	SIGMOIDE
R2 SCORE	0.96	0.65	0.89	0.89
MAE	1.98	8.07	3.54	3.54
MSE	14.94	138.77	42.08	42.08
RMSE	3.87	11.78	6.49	6.49

Se puede ver que el modelo que obtiene mejores métricas es el modelo que utiliza el *kernel* RBF, por lo que se utilizará este *kernel* para probar con los otros métodos:

LinearSVR y NuSVR. Así, nos quedaremos con el modelo de los tres que mejores resultados devuelva. La Tabla 13 muestra las métricas de cada método nombrado:

Tabla 13. Métricas de cada método de SVR

MÉTRICAS	SVR	LINEARSVR	NUSVR
R2SCORE	0.96	0.65	0.98
MAE	1.98	8.07	1.24
MSE	14.94	139.75	6.72
RMSE	3.87	11.82	2.59

4.5.3. KNN

El siguiente algoritmo que se ha probado es KNN. Este algoritmo trabaja dando un valor al punto en base a los que tiene más cerca. Los hiper parámetros utilizados para este algoritmo son los siguientes:

Número de vecinos = 5, pesos = distancia, algoritmo = auto, p = 2, métrica = minkowski

También se ha probado a buscar un modelo mejor con la función de GridSearchCV, esta función ha devuelto un modelo con los parámetros algoritmo = ball_tree y p = 1.

Se han medido los resultados de cada uno de los modelos obteniendo los resultados que se muestran en la Tabla 14:

Tabla 14. Métricas del algoritmo KNN

MÉTRICAS	MODELO A MANO	GRIDSEARCHCV
R2SCORE	0.9957	0.9957
MAE	0.30	0.30
MSE	1.73	1.76
RMSE	1.31	1.33

Las métricas de rendimiento son prácticamente la mismas, variando mínimamente en apenas tres centésimas en el error medio cuadrático, por lo que se va a optar por el primer modelo, aunque en la práctica no se va a notar la diferencia.

También se han comprobado otras métricas de rendimiento para ver los resultados que nos ofrecen, estas son las obtenidas por la validación cruzada, de la cual se ha obtenido una media de 0,9953 y una desviación estándar de 0,0007.

La Tabla 15 muestra una comparativa entre algunos valores reales calculados mediante la ecuación 47 descrita en el apartado 4.3.3 con valores predichos por el modelo de KNN que se acaba de calcular:

Tabla 15. Valor calculado frente a valor predicho por el modelo KNN

Valor calculado	Valor predicho
34,51	34,51
79,86	79,86
79,86	79,86
65,14	65,14
79,86	79,86
79,86	79,86

Se puede ver que el modelo es perfectamente capaz de predecir los valores en base a los puntos con los que se ha entrenado. Pero unos valores tan buenos pueden dar a pensar que se ha producido un sobreajuste.

4.5.4. Random Forest

Se ha probado el algoritmo de Random Forest. Para ello primero se ha probado con el algoritmo de árboles de decisión, ya que este suele ser menos costoso, pero también suele ofrecer mejores resultados. La Tabla 16 muestra una comparativa entre las métricas del mejor modelo producido por el algoritmo de Árboles de decisión y el algoritmo *Random Forest*:

Tabla 16. Comparativa de métricas entre árboles de decisión y Random Forest

MÉTRICAS	ÁRBOLES DE DECISIÓN	RANDOM FOREST
R2SCORE	0.92	0.97
MAE	2.89	1.79
MSE	33.46	11.09
RMSE	5.78	3.33

Se puede ver que la *Random Forest* ofrece mejores resultados que árboles de decisión. Este resultado era esperado ya que como hemos visto antes, *Random Forest* actúa igual que el otro algoritmo, pero con un mayor número de árboles, además de que al ser un algoritmo de aprendizaje en conjunto es más propenso a obtener mejores resultados.

4.5.5. Gradient Boosting

Otro algoritmo que se ha probado es el *Gradient Boosting*, este método se ha probado con unos parámetros de 100 estimadores de hasta tres nodos de profundidad. Con estos parámetros se han obtenido las siguientes métricas de rendimiento para el modelo generado por este algoritmo:

Para este modelo se han obtenido las siguientes métricas de rendimiento:

- Valor R2 = 0,991
- Error Medio Absoluto = 0,87
- Error Medio Cuadrático = 3,61
- Raíz del Error Medio Cuadrático = 1,9

Las métricas que ofrece son muy buenas, apenas hay error en el test.

Al probar algoritmos de descenso estocástico con los mismos parámetros, como *SGD Regressor*, las métricas son muy distintas, obteniendo un valor de r2 de 0,60, lo cual es un valor muy bajo. Esta puntuación tan mala, puede deberse a que este algoritmo realiza una aproximación lineal y como se ha visto al hacer una regresión, este problema no es un problema lineal.

4.5.6. Algoritmos de ensamblaje

Los dos últimos algoritmos que se van a probar son algoritmos procedentes de aprendizaje en conjunto, estos algoritmos son *AdaBoost* y *XGBoost*.

El primer algoritmo que se va a probar es *AdaBoost*. Tal y como se ha explicado en el apartado 2.4.5.6, este algoritmo funciona bajo un algoritmo base, que luego modifica para obtener mejores resultados, el algoritmo base sobre el que se va a trabajar en este caso es el de árboles de decisión, con un máximo de tres nodos de división. El número de estimadores por árbol es de 700, para utilizar el mismo parámetro en todos los algoritmos y ver cuál es el que mejor se optimiza. El último parámetro que queda por probar es la función de pérdida, que es en la que se basa el algoritmo para modificar los pesos con cada iteración, esta función puede ser lineal, cuadrática o exponencial, por lo que se probará el algoritmo con las tres funciones y se compararán los resultados para escoger el que tiene menor error de los tres. Así, la Tabla 17 muestra una comparativa de las métricas de rendimiento del algoritmo AdaBoost utilizando cada función de pérdida descrita anteriormente:

Tabla 17. Métricas de AdaBoost para cada función de pérdida

MÉTRICAS	LINEAL	CUADRADA	EXPONENCIAL
R2SCORE	0.93	0.93	0.66
MAE	4.52	4.79	10.97
MSE	27.15	29.88	136.90
RMSE	5.21	5.47	11.7

Se puede comprobar que el algoritmo como mejor modifica mejor los pesos es teniendo en cuenta una función de pérdida lineal. Debido a esto, se ha optado por el modelo que utiliza esta función para actualizarse. Para tener más fiabilidad, se ha probado la validación cruzada, en la que se ha obtenido una media de 0.93 y una desviación estándar de 0.01.

El segundo algoritmo que se va a probar es *XGBoost*. Tal y como se ha visto antes, este algoritmo comienza su funcionamiento creando un conjunto de árboles de decisión para, posteriormente, utilizar la técnica de Gradient Boosting con el fin de actualizar los árboles que realicen peores predicciones.

Los parámetros de los árboles de decisión son los mismos que en los casos anteriores, 700 estimadores con un máximo de profundidad de 7. La tarea de aprendizaje especificada para cada nodo era una regresión. Así, con estos parámetros se han obtenido las siguientes métricas de rendimiento:

- Valor R2 = 0,996
- Error Medio Absoluto = 0,38
- Error Medio Cuadrático = 1,58
- Raíz del Error Medio Cuadrático = 1,26

Se puede ver que este algoritmo es el que ofrece mejores resultados de todos los algoritmos que utilizan el aprendizaje en conjunto.

En el siguiente apartado, se hará una comparación de todos los algoritmos vistos a lo largo de este proyecto con el fin de ver cuál es el algoritmo que es capaz de hacer mejores predicciones.

5. RESULTADOS Y DISCUSIÓN

Tras probar todos los algoritmos vistos a lo largo de este proyecto, se han obtenido unas métricas de rendimiento para cada uno de ellos, la Tabla 18 relaciona las métricas de rendimiento de todos los algoritmos:

5.1. Métricas de rendimiento obtenidas por los modelos entrenados

Tabla 18. Comparativa de métricas de rendimiento

ID	Algoritmo	Métrica de rendimiento			
		R2 Score	MAE	MSE	RMSE
1	Regresión lineal	0,72	8.36	136,07	11.66
2	SVM	0,98	1.24	6.72	2.59
3	KNN	0,996	0,30	1,76	1,33
4	Árboles de decisión	0.92	2.89	33.46	5.78
5	Random Forest	0.97	1.37	10.80	3.29
6	Gradient Boosting	0.991	0.87	3.61	1.9
7	SGD	0.64	8.44	140.98	11.87
8	AdaBoost	0,93	4,52	27,15	5,21
9	XGBoost	0,9992	0,11	0,33	0,57

De esta tabla se pueden sacar varias conclusiones:

- La regresión lineal y *SGD Regressor* son los dos algoritmos que tienen peores métricas de rendimiento con una diferencia notable respecto al resto. Esto se puede deber a que estos dos algoritmos intentan resolver el problema de una forma lineal cuando el problema presentado no tiene este tipo de solución.
- El algoritmo *KNN* ha tenido muy buenas métricas. Esto puede ser gracias a que los accesos a las viviendas suelen estar todos muy juntos, por lo que al fijarse en la vivienda de al lado es muy fácil acertar el valor de la tuya.
- A excepción de *XGBoost*, los algoritmos de aprendizaje en conjunto no han sido los que han tenido mejores predicciones, esto puede ser debido a una falta de profundidad en sus árboles. Esta diferencia entre resultados se puede deber a

la forma de proceder de cada algoritmo, ya que *Random Forest* y *AdaBoost* no siguen aprendiendo después de llegar al final de los nodos, solo modifican los pesos dentro de estos y la forma en la que interactúan entre ellos.

- *XGBoost* es el algoritmo que obtiene mejores métricas de rendimiento de todos los algoritmos. Este era el resultado esperado ya que también es el algoritmo más potente. Esto se debe a que junta dos algoritmos de *ML* que ya obtienen muy buenas métricas por sí solos.

5.2. Implementación del mejor modelo

Tras haber probado todas las métricas de rendimiento se puede ver que *XGBoost* es el modelo que es capaz de hacer mejores predicciones de todos los analizados a lo largo de este proyecto.

Utilizando el modelo obtenido después del entrenamiento se va a predecir los valores de todos los accesos del municipio de Castellar del Vallès, con el fin de verificar si el modelo es capaz de reproducir los resultados del proceso de cálculo que se ha utilizado para obtener la calidad de vida de cada acceso.

Como se ha calculado la calidad de vida en todos los accesos del municipio para todas las edades existentes, es posible medir la calidad de la predicción del modelo, siendo estas las métricas de rendimiento de la predicción:

- Valor R2 = 0,9992
- Error Medio Absoluto = 0,11
- Error Medio Cuadrático = 0,33
- Raíz del Error Medio Cuadrático = 0,57

Se pueden ver unas métricas realmente con un error muy bajo, siendo menor de 1 y prácticamente 0. También hay que destacar que los valores obtenidos de las métricas de rendimiento son los mismos que al haber realizado el cálculo con el conjunto de datos de entrenamiento, lo que permite asegurar que no es un caso de sobreajuste

La Tabla 19 muestra una comparativa entre el valor de calidad de vida calculado con la ecuación obtenida en el apartado 4.4 y el valor predicho por el modelo obtenido de la implementación del algoritmo *XGBoost* en diez puntos elegidos.

Tabla 19. Comparativa entre los valores esperados y obtenidos en diez puntos elegidos

	LATITUD	LONGITUD	EDAD	VALOR ESPERADO	VALOR OBTENIDO	DIFERENCIA
0	41.621468	2.068474	11	52.885748	52.905148	0,0194
1	41.600270	2.085002	12	74.793875	74.718060	-0,075815
2	41.616524	2.089927	12	79.858657	79.857250	-0,001407
3	41.618820	2.089480	11	79.858657	79.858420	-0,000237
4	41.618908	2.089475	12	79.858657	79.856970	-0,001687
5	41.619307	2.089754	14	79.858657	79.861534	0,002877
6	41.619267	2.089729	10	79.858657	79.859640	0,000983
7	41.619217	2.089703	11	79.858657	79.858420	-0,000237
8	41.619142	2.089662	10	79.858657	79.859640	0,000983
9	41.619046	2.089594	10	79.858657	79.859640	0,000983

Con el fin de interpretar mejor los resultados se visualizarán en un mapa de calor donde se pueda comprobar de una forma visual (Ilustración 34) las zonas del municipio con mejor calidad de vida:

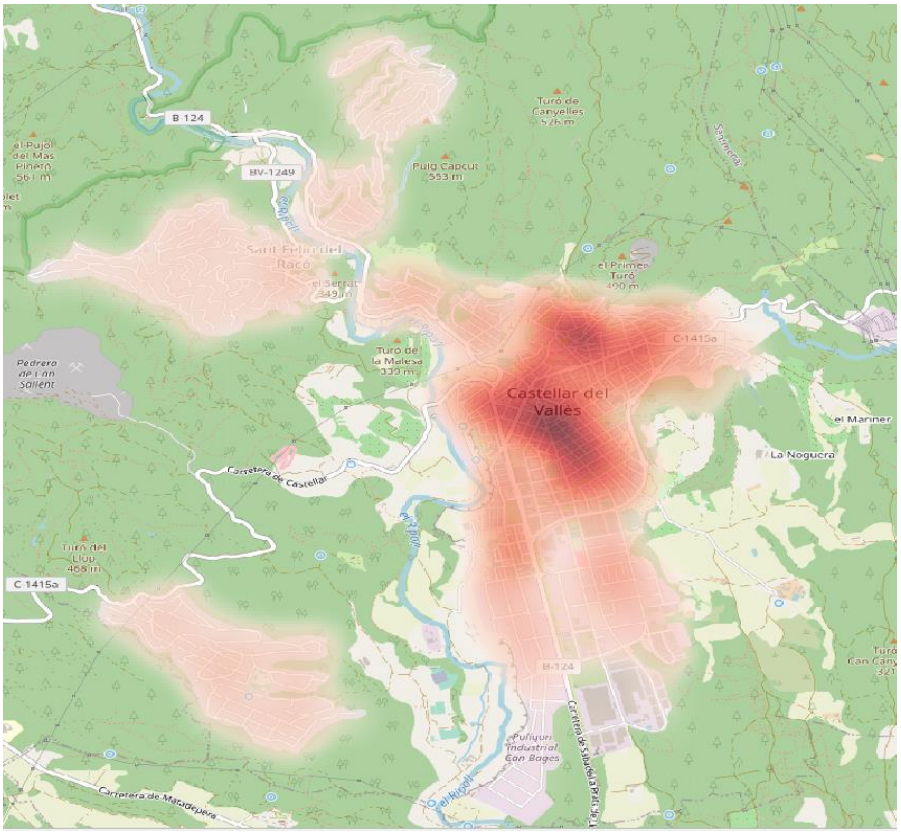


Ilustración 34. Mapa de calor de la calidad de vida del municipio de Castellar del Vallès

Fuente: Elaboración propia

Se puede ver en la Ilustración 34 como se dibuja la forma del municipio, además al observar los accesos, se aprecia que una mayor calidad de vida en la zona con mayor vivienda, siendo este el centro del núcleo de población. Para realizar una mejor comparación, se utilizará una imagen de satélite para comprobar las diferentes zonas del municipio, esta imagen se puede ver en la Ilustración 35:

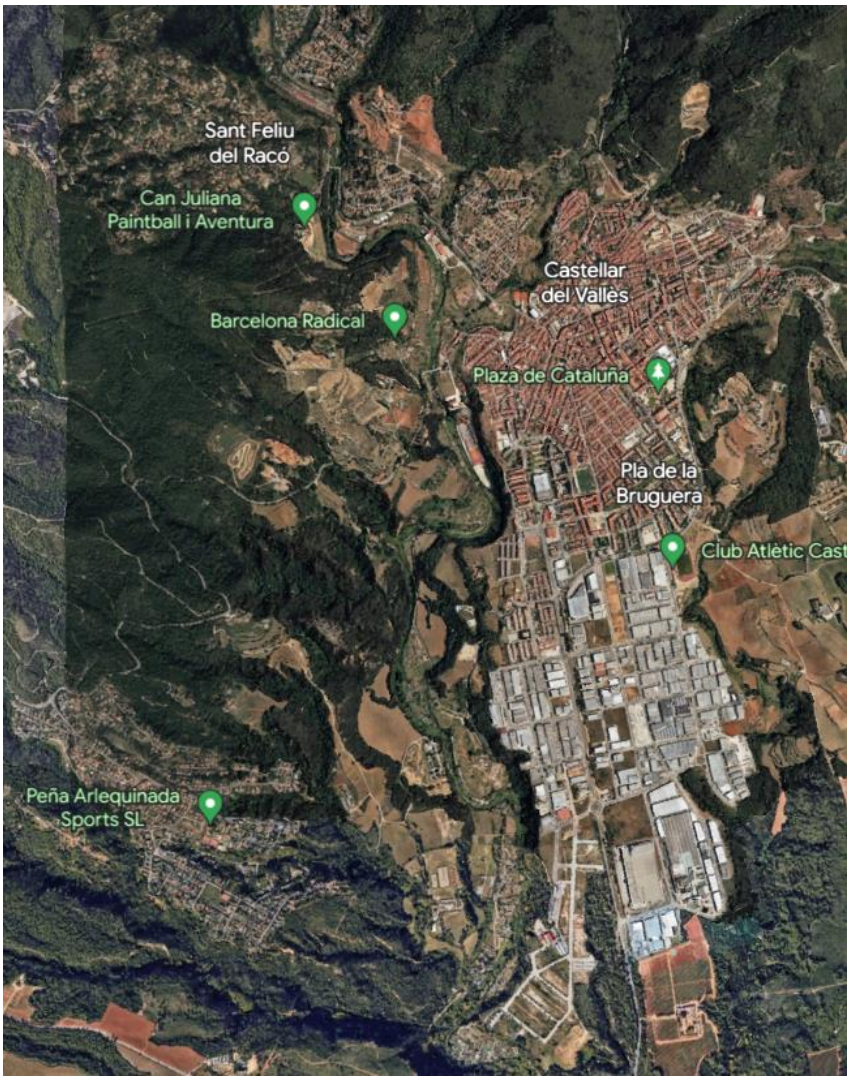


Ilustración 35. Imagen satélite del municipio de Castellar del Vallès. Fuente: [99]

Comparando la Ilustración 34 con la Ilustración 35 se puede ver como la zona del municipio con mejor calidad de vida coincide con la zona más urbana, además, también se puede comprobar que la zona con menor calidad de vida es la zona sur que es donde se concentra toda la industria.

Para ver mejor como se ajusta el mapa de calor a la realidad, la Ilustración 36 muestra el mapa de calor de Castellar del Vallès con todos los accesos a la vivienda utilizados, así se puede ver con mayor facilidad como el mapa de calor dibuja correctamente la forma del municipio:

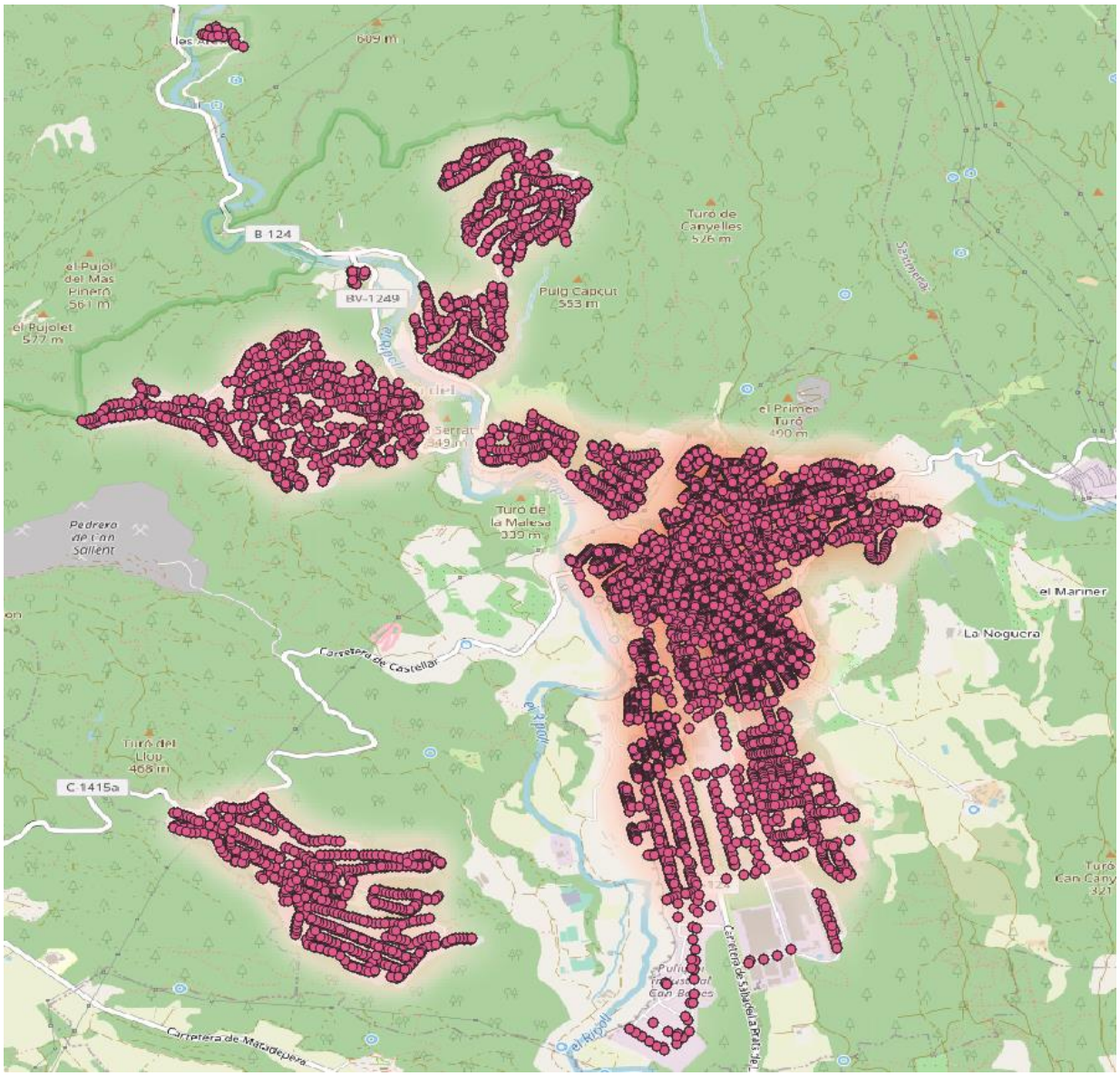


Ilustración 36. Mapa de calor de la calidad de vida del municipio de Castellar del Vallès con los accesos a las viviendas Fuente: Elaboración propia

Por último, en la Ilustración 34 se puede comprobar como la zona con mejor calidad de vida es la zona urbana del municipio, pero en esta ilustración no se puede ver cómo está distribuido, por lo tanto, la Ilustración 37 muestra el mapa de calor de esta zona urbana con un nivel de detalle mayor para que se pueda ver mejor la distribución de la calidad de vida en esta zona:

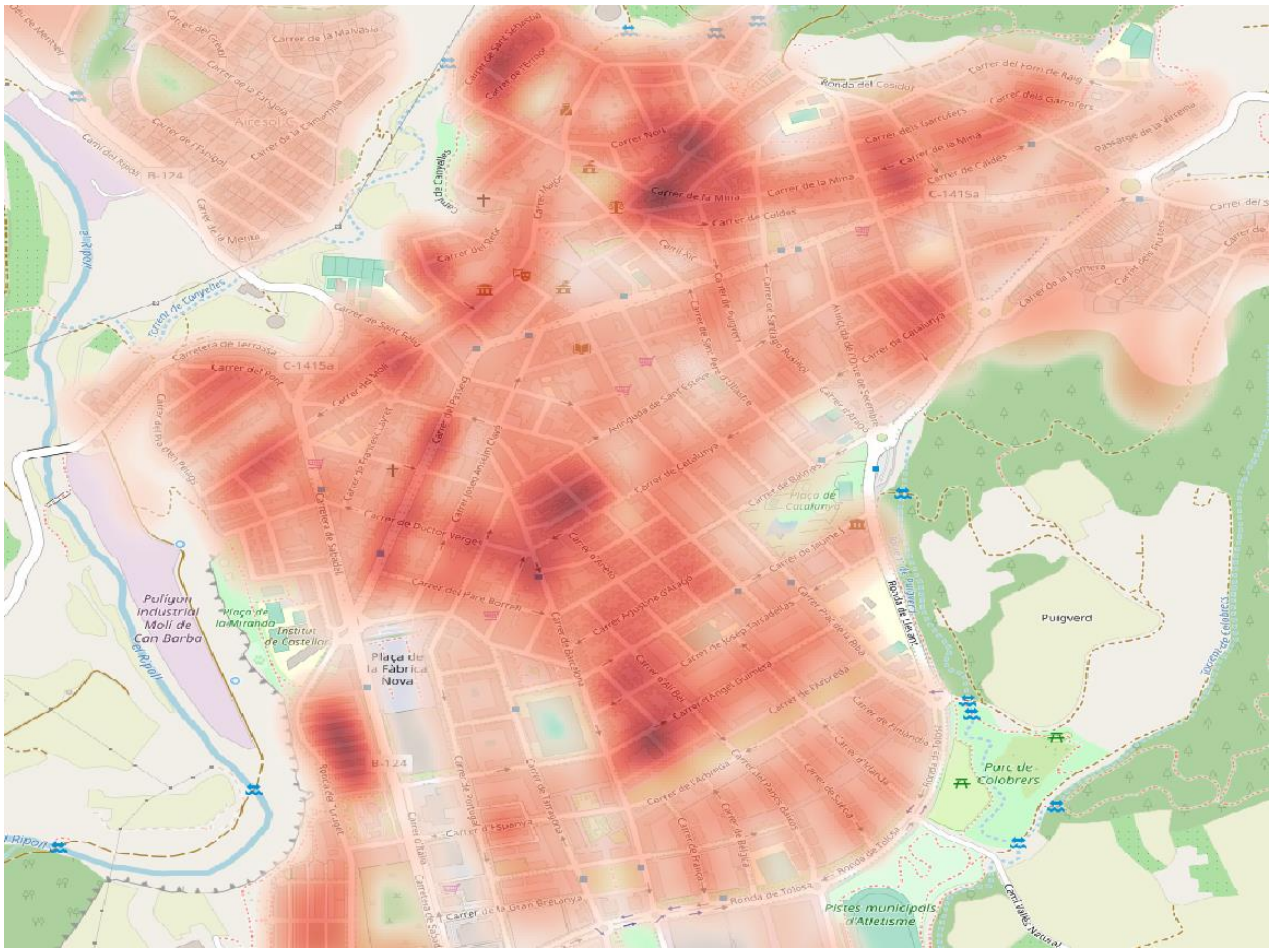


Ilustración 37. Mapa de calor de la zona urbana del municipio de Castellar del Vallès. Fuente: Elaboración propia

6. PRESUPUESTO

En este apartado se calculará un presupuesto aproximado para cubrir el coste de realización de este proyecto. Se tendrá en cuenta el coste material, que consta de software y hardware, y el coste de producción o explotación para la obtención de un presupuesto. Al final del apartado se obtendrán dos presupuestos diferentes, uno teniendo en cuenta solo el coste material, para saber cuánto costaría a una persona realizar este proyecto individualmente, y otro presupuesto incluyendo el coste de producción y explotación, para saber cuánto le costaría realizarlo a una empresa. El tiempo de realización del proyecto es la suma del tiempo de todos los puntos descritos en el apartado 0, lo que da un total de 400 horas. Los meses necesarios para realizar el proyecto se obtendrán teniendo en cuenta el número de créditos que tiene el Trabajo de Fin de Grado (TFG) y los meses que tarda en realizarse, esto se explica más detalladamente en la Tabla 20:

Tabla 20. Tiempo de Trabajo de Fin de Grado

ECTS/ TFG	h / ECTS	h/TFG	Meses
12	25	300	4

De esta forma se obtienen los meses que tarda en realizarse el proyecto mediante la operación descrita en la ecuación (49):

$$\frac{400h \times 4meses}{300h} = 5{,}33meses \tag{49}$$

De esta operación se obtiene un resultado de 5,33. Por lo que hay que sumar un mes más para poder abarcar el tiempo restante del quinto mes. Por lo tanto, para los cálculos de presupuesto se tendrá en cuenta un periodo de 6 meses.

6.1. Coste

6.1.1. Hardware

Este apartado es muy variable, ya que realmente no se necesita un hardware específico para realizar este proyecto. Dependiendo el equipo con el que se realice variará el tiempo que tardan los algoritmos en realizar sus entrenamientos, pero al no tratar con un conjunto de datos muy grande y solamente realizar cálculos numéricos no es necesario un equipo con unos componentes que destaquen. Sabiendo esto, el

presupuesto se calculará con el equipo personal que se ha realizado este proyecto, este equipo está descrito en el apartado 3.2 HARDWARECon este equipo no ha habido problemas de rendimiento, pero cabe destacar que, al ser tantos puntos, una pequeña disminución en el tiempo que tarda en calcularlos puede suponer una gran reducción de tiempo, habiendo tardado en este caso para algún proceso más de una semana. Sabiendo esto, el coste de hardware está descrito en la Tabla 21:

Tabla 21. Coste de Hardware

Material	Coste Material	Unidades	Tiempo de uso (meses)	Vida útil (meses)	Factor de amortización	Coste Total
Ordenador	1.000€	1	6	180	1/30	33,33€

6.1.2. Software

La descripción del software utilizado viene descrita en el apartado 0. Para el cálculo del coste, se dará de vida útil a la licencia de Microsoft el mismo tiempo que al ordenador personal, esto es debido a que la licencia va asociada al ordenador. Sabiendo esto la Tabla 22 describe el coste de software del proyecto:

Tabla 22. Coste de Software

Material	Coste Material	Unidades	Tiempo de uso (meses)	Vida útil (meses)	Factor de amortización	Coste Total
Licencia Windows Pro	199€	1	6	180	1/30	6,63€
Python	0€	1	6	-	-	0€
Anaconda	0€	1	6	-	-	0€
Librerías Python	0€	1	6	-	-	0€
QGIS	0€	1	6	-	-	0€
CartoCiudad	0€	1	6	-	-	0€
Microsoft Office	579€	1	6	12	1/2	289,5€
Total						296,13€

6.1.3. Producción o explotación

Al tratarse de un proyecto cuya tarea principal es el manejo de datos, la persona a contratar será un científico de datos. Al tratarse de un proyecto final de carrera, un científico de datos junior no debería tener problema en realizarlo, por lo que este es el puesto que se tendrá en cuenta para el cálculo. Sabiendo esto, el salario base promedio de un científico de datos junior es de 26.000€/año según el portal de empleo *glassdoor* [100] por lo que es el salario que se tendrá en cuenta. Además, el tiempo que estará contratado será el tiempo que dura el proyecto. Con todo esto, la Tabla 23 muestra el coste de producción o explotación asociado a este proyecto.

Tabla 23. Coste de producción o explotación

Personal	**h / Proyecto**	**Salario Anual (€/Año)**	**Salario hora (€/h)**	**Coste total**
Científico de datos	400h	26.000€	12,5€	5.000€

6.2. Presupuesto necesario

Una vez se ha calculado todo el coste se puede saber cuál es el presupuesto necesario para realizar este proyecto. Lo primero es ver cuál es el coste total, esto se puede ver mediante la Tabla 24:

Tabla 24. Coste total

Coste	**€**
Hardware	33,33€
Software	296,13€
Producción o explotación	5.000€
Total	**5.329,46€**

Este sería el coste de la realización del proyecto, pero con este presupuesto no solo no se ganaría dinero, sino que hasta se perdería, ya que hay que sumar el IVA de la venta del proyecto. Además, hay que tener en cuenta que este varía según quien sea el vendedor, descontando un 7% de Impuesto de Renta a las Personas Físicas (IRPF) si el vendedor tiene un año menos de experiencia, un 15% de IRPF si el vendedor tiene más de un año de experiencia, y no descontando nada en caso de tratarse de una empresa. Sabiendo esto, se aumentará el presupuesto un 10% para poder obtener un

beneficio, además, a este precio total se le sumará el IVA correspondiente según cada tipo de vendedor. Así, el presupuesto necesario según cada tipo de vendedor para realizar este proyecto se muestra en la Tabla 25:

Tabla 25. Presupuesto total necesario

Costes	Menos de 1 año	Más de 1 año	Empresa
Proyecto	5.329,46 €	5.329,46 €	5.329,46 €
Beneficio (10%)	532,95 €	532,95 €	532,95 €
IVA (21%)	1.231,11 €	1.231,11 €	1.231,11 €
IRPF	-410,37 €	-879,36 €	0 €
Total	**6.683,14 €**	**6.214,15 €**	**7.093,51 €**

7. CONCLUSIONES Y FUTURAS LÍNEAS DE DESARROLLO

El objetivo de este proyecto era comprobar la hipótesis de si se puede predecir la calidad de vida del municipio de Castellar del Vallès utilizando técnicas de Inteligencia Artificial. Después de haber realizado un análisis y haber entrenado suficientes modelos de Aprendizaje Automático se ha comprobado que sí es posible replicar el resultado. Por lo general, casi todos los modelos realizan predicciones bastante buenas, obteniendo en su mayoría resultados de la métrica de r2 de más de 0.95, por lo que se puede decir que predicen con bastante exactitud el resultado. Además, el algoritmo XGBoost ha obtenido resultados muy próximos a 1, de 0.9992. Estos son unas métricas de rendimiento muy buenas y en las predicciones que se han realizado, las predicciones difieren en menos de 0,1 puntos, por lo que es una diferencia que se puede obviar. Con todo esto, es posible afirmar la hipótesis de que se puede replicar el trabajo humano en este proyecto con inteligencia artificial.

La realización de este proyecto ha permitido profundizar bastante en los dos campos principales de este proyecto, el análisis geoespacial y el aprendizaje automático. Respecto al análisis geoespacial este proyecto ha resultado muy útil debido a la gran cantidad de datos que maneja, y la mayor complejidad en el proceso de limpieza de datos que eso conlleva, y las distintas formas de representarlos que hay. Respecto al campo del aprendizaje automático, este proyecto ha permitido descubrir algoritmos nuevos y formas diferentes de optimizar los algoritmos más conocidos, ya sea modificando los parámetros a mano o mediante funciones como *GridSearchCV*.

Aun así, este proyecto no ha explotado todo su potencial, ya que al ser un proyecto universitario tiene claras limitaciones de recursos, ya sea de tiempo o de presupuesto. Realizar el proyecto ha permitido ver que la hipótesis es correcta y ha destacado algunos algoritmos por encima de otros, como es el caso de XGBoost frente a la regresión lineal, por lo que una propuesta a futuro es probar estos algoritmos que han obtenido mejores resultados con un conjunto de datos más grande para ver cómo se comportan. Sería interesante que este nuevo conjunto de datos contenga más municipios ubicados en distintas localizaciones geográficas, ya que no va a tener la misma distribución un municipio costero que uno de montaña. Además, una nueva mejora al introducir nuevos municipios sería añadir más parámetros al modelo de predicción, ya que podrían influir factores que también afecten a la calidad de vida como la calidad del aire o la densidad de población. Por último, al aumentar el conjunto de datos también pasan a estar disponibles otro tipo de técnicas que pueden resultar muy interesantes, como realizar predicciones utilizando redes neuronales propias de técnicas de Aprendizaje Profundo

como se ha visto anteriormente. Esto puede resultar muy útil al aumentar el número de variables independientes ya que las redes neuronales, por lo general suelen permitir una mayor complejidad del modelo.

Todo el código utilizado en este proyecto ha sido publicado en GitHub, al cual es posible acceder a través del siguiente enlace:

Gomezzz299/TFG: Repositorio que contiene todo el código del Trabajo de Fin de Grado (github.com)

REFERENCIAS

[1] R. Costanza *et al.*, «An Integrative Approach to Quality of Life Measurement, Research, and Policy», *S.A.P.I.EN.S. Surveys and Perspectives Integrating Environment and Society*, n.º 1.1, Art. n.º 1.1, nov. 2008, Accedido: 21 de mayo de 2023. [En línea]. Disponible en: https://journals.openedition.org/sapiens/169

[2] «WHOQOL - Measuring Quality of Life| The World Health Organization». https://www.who.int/tools/whoqol (accedido 21 de mayo de 2023).

[3] U. Nations, «Documentation and downloads», United Nations. Accedido: 21 de mayo de 2023. [En línea]. Disponible en: https://hdr.undp.org/data-center/documentation-and-downloads

[4] D. Kahneman y A. Deaton, «High income improves evaluation of life but not emotional well-being», *Proc Natl Acad Sci U S A*, vol. 107, n.º 38, pp. 16489-16493, sep. 2010, doi: 10.1073/pnas.1011492107.

[5] K. Larson, «Kent Larson: Brilliant designs to fit more people in every city | TED Talk». https://www.ted.com/talks/kent_larson_brilliant_designs_to_fit_more_people_in_every_city (accedido 21 de mayo de 2023).

[6] «Introducing the 15-Minute City Project», *15-Minute City*. https://www.15minutecity.com/blog/hello (accedido 21 de mayo de 2023).

[7] C. Moreno, Z. Allam, D. Chabaud, C. Gall, y F. Pratlong, «Introducing the "15-Minute City": Sustainability, Resilience and Place Identity in Future Post-Pandemic Cities», *Smart Cities*, vol. 4, n.º 1, Art. n.º 1, mar. 2021, doi: 10.3390/smartcities4010006.

[8] «How to build back better with a 15-minute city». https://www.c40knowledgehub.org/s/article/How-to-build-back-better-with-a-15-minute-city?language=en_US (accedido 21 de mayo de 2023).

[9] «La carta en la que más de 1.000 expertos piden frenar la inteligencia artificial por ser una "amenaza para la humanidad"», *BBC News Mundo*. Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://www.bbc.com/mundo/noticias-65117146

[10] «Top Applications of Artificial Intelligence (AI) in 2023», *InterviewBit*, 5 de enero de 2022. https://www.interviewbit.com/blog/applications-of-artificial-intelligence/ (accedido 27 de mayo de 2023).

[11] I. B. School, «Quality Of Life: Everyone Wants It, But What Is It?», *Forbes*. https://www.forbes.com/sites/iese/2013/09/04/quality-of-life-everyone-wants-it-but-what-is-it/ (accedido 27 de mayo de 2023).

[12] A. Bottomley, «The Cancer Patient and Quality of Life», *The Oncologist*, vol. 7, n.º 2, pp. 120-125, abr. 2002, doi: 10.1634/theoncologist.7-2-120.

[13] L. Magee, A. Scerri, y P. James, «Measuring Social Sustainability: A Community-Centred Approach», *Applied Research in Quality of Life*, vol. 7, n.º 3, p. 239, 2012.

[14] «Research Working Papers Archive», 9 de enero de 2023. https://www.kansascityfed.org/research/research-working-papers-archive/ (accedido 21 de mayo de 2023).

[15] P. Singer *et al.*, «The Big Question: Quality of Life: What Does it Mean? How Should We Measure It?», *World Policy Journal*, vol. 28, n.º 2, pp. 3-6, jun. 2011, doi: 10.1177/0740277511415049.

[16] A. Hunter, *English: World map of countries or territories by Human Development Index scores in increments of 0.050 in 2021.* 2022. Accedido: 27 de mayo de

2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Countries_by_Human_Development_Index_(2021).svg
[17] «Human Development | Human Development Reports (HDR) | United Nations Development Programme (UNDP)», 15 de abril de 2012. https://web.archive.org/web/20120415134936/http://hdr.undp.org/en/humandev/ (accedido 27 de mayo de 2023).
[18] U. Nations, «Human Development Report 2010», United Nations, ene. 2010. Accedido: 21 de mayo de 2023. [En línea]. Disponible en: https://hdr.undp.org/content/human-development-report-2010
[19] http://www.facebook.com/chakravarthi.potharlanka, «New method of calculation of Human Development Index (HDI)», *India Study Channel*, 1 de junio de 2011. https://www.indiastudychannel.com/resources/141517-New-method-of-calculation-of-Human-Development-Index-HDI-.aspx (accedido 21 de mayo de 2023).
[20] «Home», 20 de marzo de 2023. https://worldhappiness.report/ (accedido 21 de mayo de 2023).
[21] «La felicidad, hacia un enfoque holístico del desarrollo (resolución Asamblea General) - DHpedia». https://dhpedia.wikis.cc/wiki/La_felicidad,_hacia_un_enfoque_hol%C3%ADstico_del_desarrollo_(resoluci%C3%B3n_Asamblea_General) (accedido 28 de mayo de 2023).
[22] based on version 2017 from J. Janner, *English: A detailed Robinson projection SVG map shaded by country using a distributed red and green palette according to the World Happiness Report score in 2023. Countries without data are light grey.* 2023. Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:World_map_of_countries_by_World_Happiness_Report_score_(2023).svg
[23] «Defining a New Economic Paradigm: The Report of the High-Level Meeting on Wellbeing and Happiness .:. Sustainable Development Knowledge Platform». https://sustainabledevelopment.un.org/index.php?page=view&type=400&nr=617&menu=35 (accedido 27 de mayo de 2023).
[24] «World Happiness Report.pdf». Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://www.earth.columbia.edu/sitefiles/file/Sachs%20Writing/2012/World%20Happiness%20Report.pdf
[25] «Cities and Happiness: A Global Ranking and Analysis». https://worldhappiness.report/ed/2020/cities-and-happiness-a-global-ranking-and-analysis/ (accedido 27 de mayo de 2023).
[26] G. Inc, «Understanding How Gallup Uses the Cantril Scale», *Gallup.com*, 24 de agosto de 2009. https://news.gallup.com/poll/122453/Understanding-Gallup-Uses-Cantril-Scale.aspx (accedido 27 de mayo de 2023).
[27] «Living long and living well: The WELLBY approach». https://worldhappiness.report/ed/2021/living-long-and-living-well-the-wellby-approach/ (accedido 27 de mayo de 2023).
[28] «Quality of life indicators - measuring quality of life». https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Quality_of_life_indicators_-_measuring_quality_of_life (accedido 21 de mayo de 2023).
[29] «Report by the Commission on the Measurement of Economic Performance and Social Progress - Executive summary».
[30] «LexUriServ.pdf». Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2009:0433:FIN:EN:PDF

[31] European Commission. Statistical Office of the European Union., *Final report of the expert group on quality of life indicators: 2017 edition.* LU: Publications Office, 2017. Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://data.europa.eu/doi/10.2785/021270
[32] «Productos y Servicios / Publicaciones / Publicaciones de descarga gratuita». https://www.ine.es/ss/Satellite?c=INEPublicacion_C&cid=1259937499084&p=1254735110672&pagename=ProductosYServicios%2FPYSLayout&param1=PYSDetalleGratuitas (accedido 27 de mayo de 2023).
[33] G. Pozoukidou y M. Angelidou, «Urban Planning in the 15-Minute City: Revisited under Sustainable and Smart City Developments until 2030», *Smart Cities*, vol. 5, n.º 4, Art. n.º 4, dic. 2022, doi: 10.3390/smartcities5040069.
[34] C. Patterson y L. Barrie, «Forget the conspiracies, 15-minute cities will free us to improve our mental health and wellbeing», *The Conversation*, 12 de marzo de 2023. http://theconversation.com/forget-the-conspiracies-15-minute-cities-will-free-us-to-improve-our-mental-health-and-wellbeing-200823 (accedido 27 de mayo de 2023).
[35] T. Wangchuk, «Clarence A Perry's concept of a Neighborhood Unit | Planning Tank», 23 de agosto de 2022. https://planningtank.com/planning-theory/clarence-a-perrys-neighborhood-unit (accedido 27 de mayo de 2023).
[36] «Jane Jacobs' Radical Legacy», 28 de septiembre de 2006. https://web.archive.org/web/20060928205849/http://www.nhi.org/online/issues/146/janejacobslegacy.html (accedido 27 de mayo de 2023).
[37] «Manifiesto por la reorganización de la ciudad tras el COVID-19», *ArchDaily en Español*, 18 de junio de 2020. https://www.archdaily.cl/cl/941897/manifiesto-por-la-reorganizacion-de-la-ciudad-tras-el-covid-19 (accedido 21 de mayo de 2023).
[38] *Kent Larson on Resilient Communities and Sustainability - «On Cities» Masterclass Series*, (21 de abril de 2021). Accedido: 21 de mayo de 2023. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=4QsU9DTS2rM
[39] L. D'Acci, «Simulating future societies in Isobenefit Cities: Social isobenefit scenarios», *Futures*, vol. 54, pp. 3-18, nov. 2013, doi: 10.1016/j.futures.2013.09.004.
[40] L. D'Acci, «A new type of cities for liveable futures. Isobenefit Urbanism morphogenesis», *Journal of Environmental Management*, vol. 246, pp. 128-140, sep. 2019, doi: 10.1016/j.jenvman.2019.05.129.
[41] M. Weng *et al.*, «The 15-minute walkable neighborhoods: Measurement, social inequalities and implications for building healthy communities in urban China», *Journal of Transport & Health*, vol. 13, pp. 259-273, jun. 2019, doi: 10.1016/j.jth.2019.05.005.
[42] D. Capasso Da Silva, D. A. King, y S. Lemar, «Accessibility in Practice: 20-Minute City as a Sustainability Planning Goal», *Sustainability*, vol. 12, n.º 1, Art. n.º 1, ene. 2020, doi: 10.3390/su12010129.
[43] Planning, «Plan Melbourne 2017 - 2050», *Planning*, 6 de octubre de 2021. https://www.planning.vic.gov.au/policy-and-strategy/planning-for-melbourne/plan-melbourne (accedido 21 de mayo de 2023).
[44] «Distancia euclidiana: concepto, fórmula, cálculo, ejemplo», *Lifeder*, 3 de diciembre de 2019. https://www.lifeder.com/distancia-euclidiana/ (accedido 21 de mayo de 2023).
[45] Kmhkmh, *English: euclidean distance illustration*. 2018. Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Euclidean_distance_3d_2_cropped.png
[46] «Distancia Manhattan | Visión por ordenador | PFCONA», 19 de octubre de 2021. https://pfcona.org/es/distancia-manhattan/ (accedido 21 de mayo de 2023).
[47] «File:Manhattan distance bgiu.png - Wikimedia Commons». https://commons.wikimedia.org/wiki/File:Manhattan_distance_bgiu.png (accedido 27 de mayo de 2023).

[48] «Minkowski distance - Wikipedia». https://es.abcdef.wiki/wiki/Minkowski_distance (accedido 21 de mayo de 2023).
[49] W. Pimenta, *English: Unit circles in the two-dimensional*. 2014. Accedido: 27 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:2D_unit_balls.svg
[50] R. M. Camacho, «Inteligencia artificial», Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://www.academia.edu/20634126/Inteligencia_artificial
[51] «turing.pdf». Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://redirect.cs.umbc.edu/courses/471/papers/turing.pdf
[52] D. D. Valle, *Artificial Intelligence-A Modern Approach (3rd Edition)*. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://www.academia.edu/45007883/Artificial_Intelligence_A_Modern_Approach_3rd_Edition_
[53] «¿Qué es la IA fuerte? | IBM». https://www.ibm.com/es-es/topics/strong-ai (accedido 28 de mayo de 2023).
[54] M. D. Agency, «Inteligencia Artificial vs Deep learning vs Machine Learning», *Epitech Spain*, 25 de mayo de 2022. https://www.epitech-it.es/ia-vs-deep-machine-learning/ (accedido 28 de mayo de 2023).
[55] «What is Supervised Learning? | IBM». https://www.ibm.com/topics/supervised-learning (accedido 28 de mayo de 2023).
[56] D. W. Aha, *Lazy Learning*. Springer Science & Business Media, 2013.
[57] «What is Unsupervised Learning? | IBM». https://www.ibm.com/topics/unsupervised-learning (accedido 28 de mayo de 2023).
[58] J. Brownlee, «What Is Semi-Supervised Learning», *MachineLearningMastery.com*, 8 de abril de 2021. https://machinelearningmastery.com/what-is-semi-supervised-learning/ (accedido 28 de mayo de 2023).
[59] J. Brownlee, «A Gentle Introduction to Ensemble Learning Algorithms», *MachineLearningMastery.com*, 18 de abril de 2021. https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/ (accedido 28 de mayo de 2023).
[60] «What is Bagging? | IBM». https://www.ibm.com/topics/bagging (accedido 28 de mayo de 2023).
[61] L. Breiman, «Bagging predictors», *Mach Learn*, vol. 24, n.º 2, pp. 123-140, ago. 1996, doi: 10.1007/BF00058655.
[62] «What is Boosting? | IBM». https://www.ibm.com/topics/boosting (accedido 28 de mayo de 2023).
[63] «What is Machine Learning? | IBM». https://www.ibm.com/topics/machine-learning (accedido 21 de mayo de 2023).
[64] 2uadmin, «What Is Machine Learning (ML)?», *UCB-UMT*, 26 de junio de 2020. https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/ (accedido 21 de mayo de 2023).
[65] «¿Qué es la regresión lineal?» https://es.mathworks.com/discovery/linear-regression.html (accedido 21 de mayo de 2023).
[66] Sewaqu, *English: Random data points and their linear regression. Created with the following Sage (http://sagemath.org) commands:* 2010. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Linear_regression.svg
[67] «Support Vector Machine (SVM)». https://es.mathworks.com/discovery/support-vector-machine.html (accedido 28 de mayo de 2023).
[68] «1.4. Support Vector Machines», *scikit-learn*. https://scikit-learn/stable/modules/svm.html (accedido 28 de mayo de 2023).
[69] Qluong2016, *English: Support Vector Machine model. Support vectors are created to maximize the separation between two groups.* 2016. Accedido: 28 de

mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Support_vector_machine.jpg
[70] «What is the k-nearest neighbors algorithm? | IBM». https://www.ibm.com/topics/knn (accedido 21 de mayo de 2023).
[71] «1.6. Nearest Neighbors», *scikit-learn*. https://scikit-learn/stable/modules/neighbors.html (accedido 28 de mayo de 2023).
[72] «What is Random Forest? | IBM». https://www.ibm.com/topics/random-forest (accedido 21 de mayo de 2023).
[73] «What is a Decision Tree | IBM». https://www.ibm.com/topics/decision-trees (accedido 28 de mayo de 2023).
[74] Jeremybeauchamp, *English: A visual comparison between the complexity of decision trees and random forests.* 2020. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Decision_Tree_vs._Random_Forest.png
[75] «What is Gradient Descent? | IBM». https://www.ibm.com/topics/gradient-descent (accedido 28 de mayo de 2023).
[76] T. Masui, «All You Need to Know about Gradient Boosting Algorithm − Part 1. Regression», *Medium*, 12 de febrero de 2022. https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502 (accedido 28 de mayo de 2023).
[77] «1.5. Stochastic Gradient Descent», *scikit-learn*. https://scikit-learn/stable/modules/sgd.html (accedido 28 de mayo de 2023).
[78] A. Saini, «Master the AdaBoost Algorithm: Guide to Implementing & Understanding AdaBoost», *Analytics Vidhya*, 15 de septiembre de 2021. https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/ (accedido 28 de mayo de 2023).
[79] «XGBoost Documentation — xgboost 1.7.5 documentation». https://xgboost.readthedocs.io/en/stable/index.html (accedido 28 de mayo de 2023).
[80] «What is XGBoost?», *NVIDIA Data Science Glossary*. https://www.nvidia.com/en-us/glossary/data-science/xgboost/ (accedido 28 de mayo de 2023).
[81] «Bienvenido al proyecto QGIS!» https://qgis.org/es/site/ (accedido 28 de mayo de 2023).
[82] «QGIS Project Logo», *OSGeo*, 13 de agosto de 2017. https://www.osgeo.org/projects/qgis/qgis-logo/ (accedido 28 de mayo de 2023).
[83] «Welcome to Python.org», *Python.org*, 22 de mayo de 2023. https://www.python.org/ (accedido 28 de mayo de 2023).
[84] «The Python Logo», *Python.org*. https://www.python.org/community/logos/ (accedido 28 de mayo de 2023).
[85] «Anaconda | The World's Most Popular Data Science Platform», *Anaconda*. https://www.anaconda.com (accedido 28 de mayo de 2023).
[86] «Project Jupyter | Home». https://jupyter.org/ (accedido 28 de mayo de 2023).
[87] «jupyter.github.io/share.png at master · jupyter/jupyter.github.io · GitHub». https://github.com/jupyter/jupyter.github.io/blob/master/assets/share.png (accedido 28 de mayo de 2023).
[88] «NumPy». https://numpy.org/ (accedido 28 de mayo de 2023).
[89] I. Presedo-Floyd, *English: This is a new NumPy logo.* 2020. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:NumPy_logo_2020.svg
[90] «Matplotlib — Visualization with Python». https://matplotlib.org/ (accedido 28 de mayo de 2023).
[91] Matplotlib, *English: Matplotlib logo icon*. 2015. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Matplotlib_icon.svg

[92] «pandas - Python Data Analysis Library». https://pandas.pydata.org/ (accedido 28 de mayo de 2023).
[93] The pandas development team, «pandas-dev/pandas: Pandas». 28 de mayo de 2023. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://github.com/pandas-dev/pandas
[94] «scikit-learn: machine learning in Python — scikit-learn 1.2.2 documentation». https://scikit-learn.org/stable/index.html (accedido 28 de mayo de 2023).
[95] «scikit-learn/scikit-learn». scikit-learn, 28 de mayo de 2023. Accedido: 28 de mayo de 2023. [En línea]. Disponible en: https://github.com/scikit-learn/scikit-learn/blob/42d235924efa64987a19e945035c85414c53d4f0/doc/logos/scikit-learn-logo.svg
[96] C. N. de I. Geográfica, «CartoCiudad», *CartoCiudad*. https://www.cartociudad.es (accedido 28 de mayo de 2023).
[97] «Cartociudad Web Demo». https://www.cartociudad.es/services/ (accedido 21 de mayo de 2023).
[98] «Tabla relación velocidad de peatones caminando». https://causadirecta.com/especial/calculo-de-velocidades/tablas/tabla-relacion-velocidad-de-peatones-caminando (accedido 28 de mayo de 2023).
[99] «Google Earth». https://earth.google.com/web/ (accedido 16 de junio de 2023).
[100] «Sueldo: Junior Data Scientist (Mayo, 2023)», *Glassdoor*. https://www.glassdoor.es/Sueldos/false (accedido 28 de mayo de 2023).