Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA DE SEGOVIA

Grado en Ingeniería Informática de Servicios y Aplicaciones

Predicción de Incendios Forestales mediante el estudio de factores medioambientales haciendo uso de Algoritmos de Machine Learning

---

Alumno: Carlos García-Lago Riego

Tutor: José Ignacio Farrán Martín

Predicción de Incendios Forestales mediante el estudio de factores medioambientales haciendo uso de Algoritmos de Machine Learning

Carlos García-Lago Riego

20 de junio de 2025

Índice general

Índice de figuras

Índice de cuadros

Agradecimientos

A mi familia y amigos más cercanos, a David y Ramón, y a mi tutor, José Ignacio, por ayudarme en todo momento.

Resumen

Gracias a los avances en la inteligencia artificial, actualmente se pueden estudiar problemas complejos que antes no tenían fácil solución. Este es el caso del cambio climático. El cambio climático es un problema que afecta globalmente a millones de personas. En concreto, en Castilla y León se traduce en el alargamiento de las estimaciones del verano y el invierno, y en la subida de las temperaturas. Una consecuencia de estos cambios es la mayor frecuencia e intensidad de los incendios.

Este trabajo busca estudiar los incendios forestales, analizando los factores medioambientales con el objetivo de predecir la ocurrencia de incendios. Se utilizan técnicas y modelos de machine learning para la realización del estudio. Se comienza con el estudio y tratamiento de los datos. Se continúa con la realización, entrenamiento y afinamiento de los modelos, derivando, por último, las conclusiones finales sobre los resultados obtenidos.

Palabras clave: Inteligencia Artificial, Machine Learning, Predicción de Incendios, Medioambiente, Python, Generación sintética de datos

Abstract

Due to advances in artificial intelligence, it is now possible to study complex problems that previously had no easy solution. This is the case with climate change. Climate change is a problem that affects millions of people globally. Specifically, in Castilla y León, Spain, it translates into the lengthening of summer and winter seasons, and the rise of the temperatures. One consequence of these changes is the increased frequency and intensity of fires.

This work aims to study forest fires, analyzing environmental factors with the objective of predicting the occurrence of fires. Machine learning techniques and models are used to carry out the study. It begins with the study and treatment of the data. It continues with the implementation, training and tuning of the models, deriving at the end, the final conclusions on the results obtained.

Keywords: Artificial Intelligence, Machine Learning, Fire Prediction, Environment, Python, Synthetic Data Generation

Capítulo 1

Descripción del Proyecto

1.1. Introducción

En los últimos años, el avance de la inteligencia artificial (IA) ha permitido desarrollar soluciones nuevas a problemas complejos. Una amplia gama de campos se ha beneficiado de estos adelantos tecnológicos, [1]. Uno de los problemas actuales más conocidos, y complicados, es el cambio climático. El cambio climático está provocando, como su nombre indica, cambios en los fenómenos meteorológicos a nivel global, llevando algunos a ser de carácter extremo, [4]. El aumento de las temperaturas está teniendo repercusiones en el clima mundial. En concreto, en Castilla y León, repercute en mayor frecuencia e intensidad de olas de calor, así como la prolongación del invierno y verano, [2]. Además, se ha intensificado la frecuencia y la severidad de los incendios forestales.

Esta realidad plantea un reto, anticipar con antelación cuándo y dónde podría producirse un incendio, permitiendo así una respuesta más eficiente y una posible prevención de los incendios forestales. Mediante el uso de las tecnologías de Inteligencia Artificial, se puede estudiar la predicción de los incendios forestales, [3]. La complejidad de este estudio reside en la dependencia de los incendios de múltiples factores climáticos, topográficos y humanos, así como el constante cambio de las condiciones ambientales de cada región.

Este trabajo se centra en este estudio, más concretamente, en el desarrollo de modelos de predicción de incendios forestales basados en inteligencia artificial. Dichos modelos serán capaces de analizar variables meteorológicas clave como la temperatura, la humedad, el índice de sequía o la velocidad del viento, para determinar con fiabilidad la probabilidad de ocurrencia de un incendio en una región determinada, para su posterior prevención.

1.1.1. Planteamiento del problema

Nos vamos a centrar en el creciente número de incendios forestales en Castilla y León. Se prevé que no solo crecerá el número de incendios, sino que se multiplicarán en los próximos años. De hecho, se prevé que crezcan tanto en extensión como en agresividad, [5]. Por esta razón, se necesita desarrollar un sistema de predicción de incendios que sea preciso, robusto y fiable. Tal sistema debe ser capaz de anticipar la ocurrencia de incendios forestales a partir de datos ambientales, permitiendo así implementar medidas preventivas contra los incendios.

El estudio de este problema se va a abordar mediante el estudio de datos meteorológicos como temperatura, humedad relativa, índice de sequía y velocidad del viento, entre otros. Para la predicción de incendios forestales se van a utilizar tecnologías de Inteligencia Artificial. El problema se va a estudiar como una clasificación binaria, donde se trata de determinar si, dadas ciertas condiciones climáticas, se predice que ocurra un incendio en un área determinada o no.

Existen varios métodos para estudiar este problema dentro de la Inteligencia Artificial. Entre los más destacados se encuentran los árboles de decisión, bosques aleatorios, redes neuronales y k vecinos más cercanos. En este trabajo, se explorará la aplicación de estas metodologías con el objetivo de desarrollar un sistema predictivo eficaz dentro de los recursos de los que se dispone.

En concreto, este proyecto se centra en la predicción de incendios forestales en función de variables climáticas, estudiando el problema de predicción como una clasificación binaria, con el objetivo de construir herramientas que permitan predecir y prevenir el impacto de los incendios forestales.

1.1.2. Objetivos del trabajo

Objetivo general: El objetivo general de este proyecto es el desarrollo de un modelo basado en Inteligencia Artificial (IA) que sea capaz de predecir incendios forestales a partir de factores ambientales.

Objetivos especificos:

1. Realizar un estudio exhaustivo de las distintas técnicas de IA, profundizando en las técnicas de Machine Learning (ML).
2. Procesamiento y tratamiento de los datos.
3. Entrenamiento del modelo.
4. Análisis y conclusión de los resultados obtenidos.

Restricciones:

- El alcance del proyecto está limitado por los datos obtenidos para el entrenamiento del modelo ya que son escasos. Por ello, se va a utilizar la generación de datos sintéticos para intentar paliar este problema.
- El acceso a programas como Azure Machine Learning mejoraría el manejo de los recursos y mayores opciones de análisis.

1.1.3. Estructura de la memoria

La memoria está dividida en tres secciones principales, las cuales a su vez están divididas en subsecciones:

1. Descripción del proyecto:
    - Introducción: Se presenta el contexto actual y se explica de forma clara el planteamiento del problema. Además, se acompaña de los objetivos y restricciones que existen en este proyecto.
    - Metodología: Se detallan las herramientos y métodología que se van a utilizar para el proyecto.
    - Planificación: Se realiza la estimación temporal y económica de las tareas a desarrollar, así como el balance real.
2. Marco teórico:
    - Inteligencia Artificial: Se hace una introducción de qué es y para que sirve la inteligencia artificial.
    - Ramas de la IA: Se detallan los ditintos tipos de inteligencia artificial y se ponene en su contexto histótico.
    - Pasos de un proyecto de Machine Learning: Se explican los distintos pasos que se deben tomar a la hora de hacer un proyecto, explicando el detalle las distintas subtareas y sus técnicas que se pueden aplicar.

3. Desarrollo de la propuesta y resultados:
    - Descripción del conjunto de datos: Se explica y detalla el conjunto de datos del que se dispone y que se va a estudiar.
    - Tratamiento de los Datos: Se explica las distintas trasnformaciones llevadas a cabo en su tratamiento.
    - Modelos: Se detalla los modelos elegidos para el estudio y su implementación.
    - Afinamiento de los modelos: Se realiza el entrenamiento de los modelos así como los ajustes implementados.
4. Análisis de conclusiones: Se comentan los resultados obtenidos y la causa de ellos. Además, se derivan las conclusiones donde se hace un balance del proyecto.
5. Trabajo Fututo: Se comentan las futuras líneas de Investigación.
6. Bibliografía: Se incluyen todo el material utilizado para la realización del proyecto.

1.2. Metodología

En esta sección, se va a presentar de forma detallada la metodología escogida y las distintas herramientas que se han utilizado para la realización de este proyecto.

1.2.1. Metodología

La metodología que se va a utilizar es la tradicional o predictiva. Más concretamente, se va a utilizar el modelo en cascada. Esta parte se basa en [6] y [7].

La metodología tradicional en cascada utilizada en la gestión de proyectos se basa en una estructura secuencial preestablecida que sigue las siguientes fases: Inicio, Planificación, Diseño, Implementación, Pruebas, Funcionamiento y Mantenimiento. Estas etapas se realizan de arriba hacia abajo, desarrollando sus tareas en bloques independientes y en este orden. Cada fase debe completarse y revisarse antes de avanzar a la siguiente.

Este enfoque destaca por la necesidad de una buena documentación, la planificación anticipada y el control del proceso. Esta metodología se apoya en estándares como el PMBOK y tiene como ventajas la definición clara de objetivos, procesos controlables, documentación estructurada y una mayor responsabilidad en la gestión.

1.2.2. Herramientas

El proyecto ha requerido la integración de varios recursos y herramientas, las cuales se detallan y explican a continuación:

- **PyCharm:** PyCharm es un entorno de desarrollo integrado (IDE) especializado en Python, desarrollado por la empresa JetBrains, [8]. Esta plataforma es ampliamente utilizada para la creación de aplicaciones en Python y es compatible con los principales sistemas operativos: Windows, Linux y macOS. En el marco de este proyecto, se ha empleado el lenguaje de programación Python. Python es un lenguaje de alto nivel, orientado a objetos, que se caracteriza por una sintaxis clara, concisa y legible. En concreto, se ha sido utilizado en este trabajo tanto para la implementación de distintos modelos de inteligencia artificial como para el procesamiento, análisis y visualización de datos.
- **Librerías:** Se han utilizado las siguientes librerías en el desarrollo de los modelos y el procesamiento de los datos: NumPy y pandas, empleadas para

la manipulación y análisis eficiente de estructuras de datos y grandes volúmenes de información. scikit-learn, una librería esencial para el desarrollo y evaluación de modelos de machine learning supervisado y no supervisado. TensorFlow y Keras, herramientas fundamentales para la creación y entrenamiento de modelos de aprendizaje profundo, gracias a su capacidad para trabajar con redes neuronales complejas y grandes conjuntos de datos.

- **LaTeX:** LaTeX es un sistema de composición de textos diseñado para la creación de documentos científicos, técnicos y académicos que incluyen fórmulas matemáticas complejas. Está basado en un conjunto de macros del lenguaje TeX, cuyo objetivo es simplificar el uso de esta potente herramienta tipográfica, [9]. Gracias a su precisión y calidad en la maquetación, LaTeX se ha convertido en el estándar para la redacción de artículos académicos, tesis y libros técnicos, ofreciendo un acabado tipográfico comparable al de las editoriales científicas profesionales. En el desarrollo de este proyecto se ha utilizado Overleaf, una plataforma online que permite redactar documentos en LaTeX directamente desde el navegador. Overleaf no solo facilita la escritura y visualización del contenido en tiempo real, sino que también ofrece funcionalidades colaborativas, como la edición simultánea por varios usuarios, el control de versiones, [10]. Esta herramienta ha sido empleada en concreto para la redacción de la memoria del proyecto.

- **UC Irvin Machine Learning Repository:** Se trata de un repositorio de datos online que ofrece una extensa colección de datasets reales. Estos conjuntos de datos están disponibles de forma pública y gratuita. Muchos de estos datasets están además acompañados de descripciones detalladas, documentación técnica o metadatos que facilitan su comprensión y uso adecuado. En el contexto de este proyecto, se han extraído de dicho repositorio los datos utilizados para el entrenamiento del modelo, [52].

- **Outlook:** Outlook 365 es una aplicación de correo electrónico y gestión de información personal que forma parte de Microsoft 365, [11]. Se trata de una herramienta basada en la nube que permite la gestión del correo electrónico, el calendario, la agenda de contactos, la planificación de tareas y la organización de reuniones. En el contexto de este proyecto, Outlook 365 ha sido la herramienta empleada para la comunicación con el tutor académico.

1.3. Planificación

La siguiente sección se va a realizar la planificación temporal y económica, así como se hará un balance entre la estimación original y la realidad.

1.3.1. Estimación Temporal

La planificación inicial del periodo de desarrollo de este proyecto abarca desde el 24 de febrero hasta el 20 de junio. En dicho marco temporal, se van a calcular las horas totales que conlleva el proyecto. Para ello, se va a utilizar una EDP(Estructura de Desglose de Trabajo) en la que vamos a dividir el proyecto en los diferentes objetivos y sus tareas. Posteriormente, se realizará una estimación de las horas de cada tarea de la EDP utilizando PERT (Program Evaluation and Review Technique) para dicha estimación. Finalmente, se representará la relación y secuencialización de las tareas mediante el uso de un diagrama de Gantt.

EDP

A continuación, se van a especificar los objetivos en los que se divide el proyecto, así como una explicación de sus tareas necesarias para poder alcanzar dichos objetivos:

- Investigación del estado del arte: Este objetivo se corresponde con todas las tareas de investigación, estudio y obtención de información previa a la realización del proyecto. Aquí, se van a llevar a cabo las tareas de investigación de los conceptos básicos y de los proyectos de ML. También, se analizaran los distintos modelos y se selccionan los que se van a implemnetar. Finalemente, se estudia las diferentes tecnologiás que hay a nuestra disponisión y se elije y prepara el entorno en el que se va a desarrollar el estudio. La mayor parte de este objetivo es la lectura de artículos y libros donde seleccionare la información relevante para mi proyecto.

- Estudio de los datos: La primera tarea es buscar repositorios de datos para el proyecto. Se buscan repositorios públicos con capacidad para poder realizar un análisis sobre ellos. Un vez elejido los conjuntos de datos que se van a usar, se estudian las distintas características de los datos así como se limpian y transforman los datos para poder realizar su posterior estudio.

- Construcción y entrenamiento del modelo: Aquí, nos centramos en la implementación de los distinstos modelos previamente elejidos y su ajuste. Para ello, una vez construido el modelo, se entrena con los datos y se evaluan los resulatdos obtenidos. Después, se corrijen los errores y se realzia los ajustes

necesarios. Posteriormenete, se vuelve a entrenar el modelo y se evaluan los resultados nuevos. Este proceso se repetirá varias veces.

- Análisis y presentación de los resultados: Este objetivo consiste en analizar los resultados y sacar las conclusiones del estudio realizados. Se da por finalizado el entrenamiento y se analizan los resulatdos obtenidos así como las causas y factores de estos resultados. En este objetivo, es donde se introducen las tareas de documentar el proceso llevado a cabo y redactar la memoria, aunque esto es una tarea que se realiza a lo largo de todo el proyecto.

La representación gráfica de la EDP explicada se encuentra a continuación:

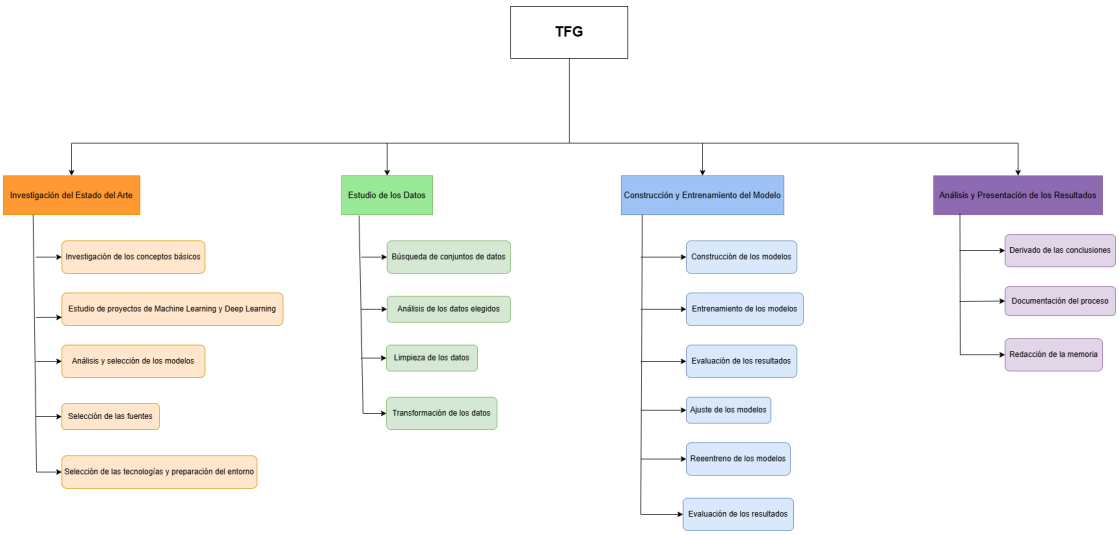

Figura 1.1: EDP que describe los objetivos del proyecto y las tareas en las que se dividen

PERT

EL siguiente paso en la planificación temporal es la estimación horaria de cada tarea. Para dicha estimación, vamos a utilizar el método PERT, [15]. Este método consiste en estimar la duración de cada tarea del proyecto. Considera el tiempo de cada tarea como una variable aleatoria que sigue una distribución Beta. Para la estimación de cada tarea se utiliza el promedio ponderado de tres estimaciones diferentes:

- T_m: estimación basada en una evaluación realista para completar el trabajo y los costes previstos.

- T_o:la estimación se determina mediante un análisis del mejor escenario para la actividad, considerando todas las condiciones favorables.

- T_p: la estimación se basa en un análisis del peor escenario para la actividad.

La estimación se obtiene a través de la siguiente fórmula,

$$T_E = \frac{T_o + 4T_m + T_p}{6}]$$

Con desviación estándar,

$$\sigma = \frac{T_o - T_p}{6}$$

La estimación temporal es la siguiente

Tarea	T_o	T_m	T_p	T_E
Investigación del estado del arte				
Investigación de conceptos básicos	10	12	20	13,00
Estudio de proyectos de Machine Learning y Deep Learning	30	40	55	40,83
Análisis y selección de los modelos	20	35	45	34,16
Selección de las fuentes	20	25	30	25,00
Selección de las tecnologías y preparación del entorno	10	15	18	14,66
Estudio de los Datos				
Búsqueda de conjuntos de datos	10	12	15	12,16
Análisis de los datos elegidos	5	8	10	7,83
Limpieza de los datos	10	12	15	12,16
Transformación de los datos	15	25	30	24,16
Construcción y Entrenamiento del Modelo				
Construcción de los modelos	40	50	60	50,00
Entrenamiento de los modelos	2	3	5	3,16
Evaluación de los resultados	3	4	5	4,00
Ajuste de los modelos	10	15	20	15,00
Reentreno de los modelos	2	3	5	3,16
Evaluación de los resultados	3	4	5	4,00
Análisis y Presentación de los Resultados				
Derivado de las conclusiones	8	10	12	10,00
Documentación del proceso	20	22	25	22,16
Redacción de la memoria	30	35	45	35,83
			Tiempo estimado total:	**331,27**

Cuadro 1.1: Cálculo de la estimación del tiempo de las tareas del proyecto

La estimación que hemos obtenido es de 331,27 h. Dado que un Trabajo Fin de Grado consta de 12 créditos ECTS y cada crédito ECTS equivale a 25 horas de trabajo según la Universidad de Valladolid, se deberían dedicar 300 horas a un TFG. Como podemos comprobar, la estimación temporal supera en 31 horas dicha cifra.

Diagrama de Gantt

Para concluir con la estimación temporal del proyecto, se va a ilustrar el diagrama de Gantt que permite presentar de forma gráfica la planificación por semanas de los objetivos y de las tareas del proyecto.

Partimos de que el proyecto comienza el 24 de febrero y finaliza el 20 de junio. En la parte superior, podemos observar representadas las semanas. En total, 17 semanas. En la parte izquierda del diagrama, se pueden observar las distintas tareas organizadas por objetivos. En el diagrama, podemos apreciar la organización semanal de cada tarea, así como la duración de cada objetivo representado en distintos colores.

- Naranja: Representa las tareas que tienen como objetivo la investigación del estado del arte.
- Verde: Se corresponde con las tareas que se realizan de acuerdo al objetivo de estudio de datos.
- Azul: Las tareas representadas son las que se realizan en el objetivo construcción y entrenamiento del modelo.
- Morado: Estas tareas corresponden al objetivo de análisi y presentación de resultados.

El diagrama comienza con la etapa de investigación durante las primeras semanas, continuando con el estudio de los datos y la construcción y entrenamiento del modelo. Finalmente, las últimas semanas se corresponden con el análisis de resultados y redacción de la memoria. Se puede observar que las tareas de documentación del proceso y redacción de la memoria se realizan durante la mayor parte de la duración del proyecto. Esto se debe a la necesidad de documentar cada tarea del proyecto.

Cabe destacar que la duración de las tareas es aproximada en cuanto a las semanas. De esta manera, se pueden visualizar más cómodamente en el diagrama.

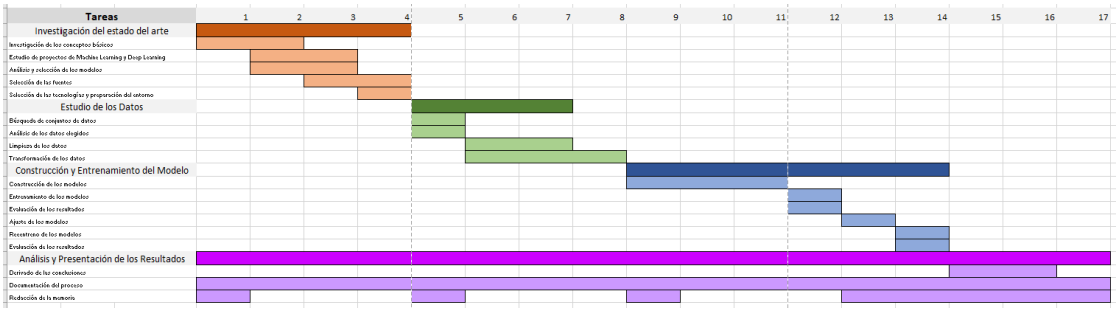

Figura 1.2: Diagrama de Gantt que descreibe las distintas tareas del proyecto y las semanas en las que se estima realizarlas.

1.3.2. Presupuesto económico

La estimación del presupuesto económico se va a dividir en dos secciones: los recursos técnicos y los recursos humanos. Los recursos técnicos representan el equipo y las herramientas, tanto software como hardware, que se van a necesitar para el desarrollo del proyecto. Los recursos humanos se refieren al coste de los trabajadores involucrados en el proyecto.

Recursos Técnicos
En cuanto al hardware, el proyecto se ha desarrollado en un ordenador portátil, modelo HP Pavlion Gaming Laptop 15-dk0xxx. Tiene una CPU Intel Core i7-9750H y 16 GB de memoria RAM. Con una vida ítil aproximadamente de 6 años y un precio de 950 €, y teniendo en cuenta que la duración del proyecto es de unos 4 meses aproximadamente, su coste es

$$950.\frac{4 \quad meses}{12 \quad meses,6 \quad años} = 52,77€$$

La conexión a Internet cuesta 30 € al mes, luego para los 4 meses de proyecto tenemos

$$30,4 \quad meses = 120€$$

El total de estimación para el hardware es de 172,77 €.

En cuanto al software, todos los programas y aplicaciones utilizadas son gratuitas. Por lo que no tienen repercusión en el presupuesto.

Recursos Humanos
Para la estimación económica de los recursos humanos se va a hacer una estimación del coste de cada hora en función del puesto de trabajo al que se adscribe la tarea que se realiza. Analizando los distintos objetivos y sus tareas, podemos dividir los

objetivos en dos perfiles de trabajo: Investigador de datos y Data Scientist [12]. Un investigador de datos se encarga de la investigación relacionada con los datos. Un Data Scientist es un perfil que transforma, analiza, representa los datos haciendo uso de técnicas de Machine Learning, entre otras. Por lo tanto, investigador de datos se corresponde con el primer objetivo, investigación del estado del arte. Los tres objetivos restantes, estudio de los datos, construcción y entrenamiento de los modelos y análisis y presentación de resultados, entran dentro del perfil de Data Scientist. Luego, las primeras 127.65 horas, se corresponden con el rol de investigador de datos, las 203.62 horas restantes, se relacionan con el perfil de Data Scientist.

Ahora, tomemos una estimación del salario de cada rol y calculemos el coste por hora. Una estimaicón del salario base del rol de investigador del dato es de entre 25.000 y 42.000 €. Dado mi poca experiencia laboral, se va a estimar 25.000 € para este rol, [13]. Para un Data Scientist junior sin experiencia, se estima un salario de entre 30.000 y 40.000 €, [14]. Sin embargo, dado mi poca experiencia, se va a estimar unos 30.000 €. Al mes, se estiman 21 días laborales, así como 8 horas diarias. Dado que se suponen 14 pagas anuales, obtenemos que un investigador de datos recibe 10.63 €/h y un Data Scientist recibe 12.76 €/h. Por lo tanto, el coste total de los recursos humanos asciende a

$$127,65\text{h}{,}10,63€/\text{h} + 203,62\text{h}{,}12,76€/\text{h} = 3955,11€$$

1.3.3. Presupuesto total

Teniendo en cuenta los costes del Hardware, Software y recursos humanos, obtenemos unos costes totales de 4.127.88 €.

	Hardware	**Software**	**RRHH**	**Total**
Estimación	172.77 €	0.00 €	3955.11 €	4127.88 €

Cuadro 1.2: Estimación del presupuesto total del proyecto.

1.3.4. Balance

A continuación, vamos a analizar las diferencias entre las estimaciones realizadas al inicio del proyecto y la realidad llevada a cabo a lo largo del proyecto. Este análisis se ha hecho al terminar el proyecto.

Balance temporal
En esta sección, vamos a comparar la duración de las tareas estimadas y reales, así como la cronología estimada y la real. Comencemos analizando la duración de

las tareas. A continuación, se presenta una tabla comparativa de las tareas con sus tiempos estimados y reales.

Tarea	T_E	Tiempo real
Investigación del estado del arte		
Investigación de conceptos básicos	13,00	10
Estudio de proyectos de Machine Learning y Deep Learning	40,83	45
Análisis y selección de los modelos	34,16	20
Selección de las fuentes	25,00	15
Selección de las tecnologías y preparación del entorno	14,66	15
Estudio de los Datos		
Búsqueda de conjuntos de datos	12,16	15
Análisis de los datos elegidos	7,83	10
Limpieza de los datos	12,16	10
Transformación de los datos	24,16	25
Construcción y Entrenamiento del Modelo		
Construcción de los modelos	50,00	60
Entrenamiento de los modelos	3,16	6
Evaluación de los resultados	4,00	6
Ajuste de los modelos	15,00	12
Reeentreno de los modelos	3,16	6
Evaluación de los resultados	4,00	6
Análisis y Presentación de los Resultados		
Derivado de las conclusiones	10,00	10
Documentación del proceso	22,16	20
Redacción de la memoria	35,83	42
Total	**331,27**	**332**

Cuadro 1.3: Comparación entre tiempo estimado y tiempo real de las tareas del proyecto

Como se puede observar, en general, las estimaciones iniciales se han cumplido a lo largo del proyecto, siendo el tiempo total el mismo número de horas que el estimado. Lo que quiere decir que se ha respetado la estimación inicial. Comentemos ahora las tareas con más discrepancias:

- Análisis y selección de los modelos: El tiempo estimado de esta tarea es de 14 horas más que el tiempo real. Esto se debe a que en la realidad, una vez investigados los conceptos básicos junto con los proyectos de Machine Learning y Deep Learning, ya tenía una idea bastante clara de los conceptos y de los modelos. Por lo tanto, a la hora de investigar y seleccionar los modelos elejidos ya entendía cómo funcionaban y qué diferencias existían entre uno y otro.

- Selección de las fuentes: la selcción de las fuentes utilizadas para el estuido del proyecto ha sido de 10 horas menos que el tiempo estimados, ya que he utilizidado la misma fuente para distintos temas relacionados con el estudio. También se han usado fuentes reconocidas. Algunas de estas fuentes utilizadas son IBM, Microsoft, Medium y diversas universidades.

- Construcción de los modelos: La diferencia es de 10 horas más en la realidad respecto al tiempo estimado. La construcción de los modelos y sus métricas de evaluación ha llevado más tiempo del estimado. En la estimación inicial no se tuvo en cuenta la implementación de las métricas de evaluación ni alguna parte del código necesaria para el correcto entreno de los modelos.

Continuemos con la estimación temporal reflejada en el diagrama de Gantt. Se va a comparar la duración temporal estimada inicialmente para cada objetivo y las semanas reales que se han utilizado. En general, se ha seguido el diagrama a lo largo del proyecto. El único cambio que se ha producido es que se ha utilizado una semana más para la construcción y entrenamientos de los modelos, y por tanto, se ha utilizado una semana menos en el objetivo de análisis y presentación de resultados. La razón es que la construcción de los modelos se ha alargado una semana más. En cuanto a la estimación de finalización, se ha cumplido, terminando así el proyecto el 20 de junio. El retraso de la tarea de construcción de los modelos no ha supuesto que se alargue una semana más el modelo.

Balance económico

En cuanto al balance económico, no ha habido discrepancias con la estimación inicial. En cuanto al hardware usado, ha sido el planteado inicialmente, por lo que no ha variado el coste inicial. Respecto al balance económico correspondiente a los recursos humanos, hay un pequeño desajuste. Esto se debe a que la estimación inicial suponía un tiempo total de 331,27 horas,, siendo el tiempo total final de 332

horas. Como este desajuste se debe a tareas de construcción de los modelos, se cataloga dentro de tareas realizadas por un Data Scientist. Por lo tanto, recalculamos el presupuesto acorde a este cambio:

$$127,65\text{h}{,}10,63€/\text{h} + 204,35\text{h}{,}12,76€/\text{h} = 3964,43€$$

Esto hace que el presupuesto aumente en $9,31€$. Finalmente, el presupuesto total es de $4137,19€$. Se observa un pequeño ajuste poco significativo con respecto a la estimación inicial.

Capítulo 2

Marco teórico

En este capítulo, se presentan las bases teóricas que se necesitarán para el desarrollo del proyecto. Se abordarán los conceptos de Inteligencia Artificial, Machine Learning, Deep Learning y Redes neuronales. El objetivo es proporcionar un marco teórico claro sobre el que se basará el proyecto.

2.1. Inteligencia Artificial

Actualmente, cuando se hace referencia a la Inteligencia Artificial, a todos se nos viene a la cabeza aplicaciones como ChatGPT o Gemini. Aunque estas aplicaciones se basan en la IA, son solo una rama muy concreta de la IA, la IA generativa. La Inteligencia Artificial engloba muchas más ramas con diferentes y variadas utilidades y aplicaciones. Esta sección se basa en [16], [17] y [18].

La Inteligencia Artificial se define como la tecnología que combina el conjunto de algoritmos, procesos y sistemas, encargados de desarrollar programas que imitan la inteligencia humana y sus capacidades de aprendizaje, comprensión, resolución de problemas, toma de decisiones, creatividad y autonomía. Una IA es una máquina con inteligencia comparable a la humana, capaz de resolver cualquier tipo de problema y de adaptarse a cualquier entorno. Se trata de algoritmos capaces de aprender, actuar y adaptarse a diferentes contextos. Lo hacen a través del estudio, asimilación y aprendizaje de información de acciones que realizan los seres humanos. Por tanto, tiene como objetivo replicar la inteligencia humana en las máquinas. Busca la mejora de los procesos sin sustituir la inteligencia humana.

Para la IA es imprescindible que se relacione con el entorno específico en el que se desarrolla para que aprenda y actúe para resolver un problema específico. Su comportamiento se basa en todo lo que aprenden de lo que recopilan de su entorno

a lo largo del tiempo. En función del contexto hay dos tipos de IA:

- La Inteligencia Artificial estrecha o débil. Esta inteligencia se corresponde con los sistemas de IA diseñados que se encragan de desempeñar una tarea específica o un conjunto de tareas. Solo funciona dentro de un contexto limitado y su objetivo es realizar la tarea de forma eficiente. Sin embargo, a pesar de su aparente inteligencia, esta IA esta limitada a imitar. Algunos ejemplos de IA estrecha son: el motor de búsqueda de Google, el software de reconocimiento de imágenes, los asistentes virtuales como Siri o Alexa, los vehículos autónomos, o el sistema Watson de IBM. .

- La Inteligencia Artificial general. Esta inteligencia posee la capacidad de comprender, aprender y aplicar conocimientos en una amplia gama de tareas simulando una inteligencia igual a la humana. Puede adaptarse a cualquier contexto y a cualquier tipo de problemas. Actulamente, ste nivel de IA es teórico.

Según la Comisión Europea, también podemos clasificar la IA en dos tipos:

- La IA Software: Este tipo busca un sistema de programación que realice e imite tareas que realiza ser humano. Tiene como obejtivo facilitar la realización de tareas. Algunos ejemplos son asistentes virtuales, análisis de imágenes o sistemas de reconocimiento de voz y facial.

- La IA integrada: Se refiere a productos de uso cotidiano cuya IA se implemente para desempeñar funciones que nos permiten ser más eficaces. En esta categoría se cuentran los drones, vehículos autónomos o el Internet de las cosas.

La IA ofrece una variedad de beneficios en muchos sectores. Uno de sus principales beneficios es la automatización de tareas repetitivas, permitiendo así a las personas realizar actividades de mayor valor añadido. Además, la IA ayuda con la toma de decisiones, aportando predicciones rápidas y precisas, pudiendo procesar grandes volúmenes de datos. También reduce la incidencia de errores humanos y fallos potenciales. Otro beneficio es su disponibilidad continua, garantizando un rendimiento constante. La IA también reduce riesgos físicos al poder reemplazar trabajos peligrosos para los humanos.

Muchos son los usos en los que la IA se puede emplear. Algunos de ellos son: la automatización de procesos, la detección de patrones, proporcionar información personalizada a partir del historial del cliente, y la optimización de procesos. La IA también ayuda con el desarrollo y modernización de software utilizando

herramientas que automatizan la escritura de código y la migración a otras aplicaciones. En definitiva, la IA se ha convertido en una herramienta fundamental para la transformación digital de las organizaciones.

2.2. Ramas de la IA

La Inteligencia Artificial comenzó en 1950, cuando el matemático Alan Turing se preguntó si las máquinas podría pensar. El artículo que presentó Turing, Computing Machinery and Intelligence, junto con el Test de Turing, sentaron las bases de la inteligencia artificial. En el artículo se identifican cuatro enfoques principales que han definido el campo de la inteligencia artificial: el pensamiento humano, el pensamiento racional, la acción humana y la acción racional. Los dos primeros se centran en el razonamiento y el pensamiento de las máquinas, mientras que los otros dos están orientados al comportamiento. Basándose en parte en estos enfoques, han surgido distintas ramas dentro de la Inteligencia Artificial. Esta sección se basa en [16], [17], [18] y [19].

Como se puede ver, la IA existe desde hace mucho tiempo, pero en el último lustro es cuando ha experimentado mayor desarrollo. Actualmente, podemos encontrar IA en muchas aplicaciones en la vida diaria. La mayoría de investigaciones de IA se centran en los avances en IA generativa, una tecnología que puede crear textos, imágenes, vídeos y otros contenidos. Para comprender la IA generativa, es importante comprender primero las tecnologías sobre las que se basan, que son Machine Learning (ML) y Deep Learning.

A continuación, se describen las principales ramas en las que se divide la Inteligencia Artificial. Estas reflejan su evolución a lo largo del tiempo, pudiendo ver como a partir de las ramas que han ido surgiendo, surgían otras nuevas. La primera en desarrollarse fue el Machine Learning, que sentó las bases para las ramas posteriores. A partir de esta, surgió el Deep Learning, y más recientemente ha nacido la Inteligencia Artificial Generativa. La relación jerárquica entre estas ramas se muestra en la siguiente imagen.

2.2.1. ML

Por debajo de la Inteligencia Artificial se encuentra el Machine Learning(ML) o aprendizaje automático. El aprendizaje automático engloba todas las técnicas que permiten a las máquinas aprender de los datos y realizar una tarea específica, sin necesidad de estar programados para estas tareas específicas. En otras palabras, consiste en proporcionar datos a un sistema para que aprenda de los datos y realice

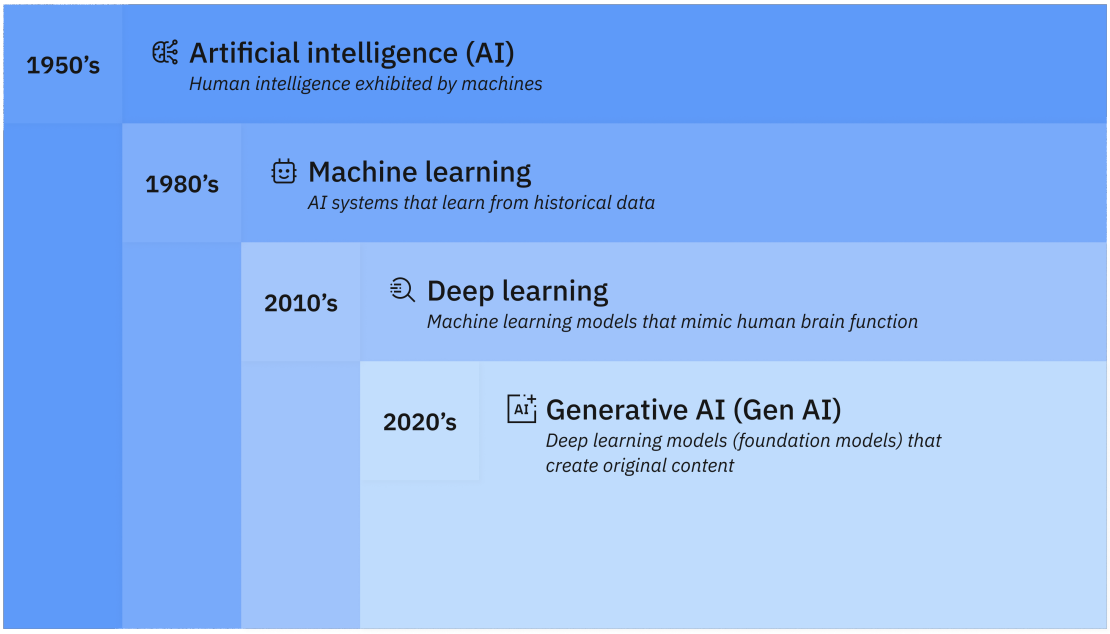

Figura 2.1: Ilustración de las ramas anidadas de la IA que contiene cuándo surgieron y una breve explicación de cada una.

inferencias a partir de ellos.

El Machine Learning contiene dos tipos:

- Supervisado: El aprendizaje supervisado solo trabaja con datos etiquetados. Esto quiere decir que los datos han sido procesados y que por lo tanto han recibido intervención humana antes de ser usados por el modelo. En el aprendizaje supervisado, los humanos emparejan cada ejemplo de entrenamiento con una etiqueta de output. De esta manera, el modelo aprende la asignación entre entradas y outputs en los datos de entrenamiento, y así puede predecir las etiquetas de los datos nuevos.
- No supervisado: Este aprendizaje trabaja con datos sin etiquetar. Los datos con los que se entrenan los modelos de datos sin procesar. Por tanto, los datos no han tenido intervención humana previa.

También existe otro tipo dentro del Machine Learning: aprendizaje por refuerzo. Este tipo de aprendizaje implica que el sistema recibe retroalimentación en base al análisis y conclusiones que ha realizado, permitiendo el ajuste de decisiones incorrectas.

Existen muchos algoritmos de Machine Learning, algunos de ellos son: regresión

lineal, regresión logística, árboles de decisión, bosques aleatorios, máquinas de vectores de soporte (SVM), k vecinos más cercanos (KNN) o clustering. Cada algoritmo tiene distintas bases matemáticas y se usan para distintos tipos de problemas. Su elección depende de los datos a estudiar y del objetivo del estudio.

Otro algoritmo de Machine Learning que está en auge ahora son las redes neuronales tradicionales o Neural Networks. Las redes neuronales se construyen con el objetivo de imitar la estructura y función del cerebro humano. Esta dividida en capas interconectadas de nodos, que se corresponden con las neuronas. Estas capas o neuronas trabajan juntas para procesar y analizar datos. Son muy útiles para tareas como identificar patrones y relaciones en grandes cantidades de datos.

2.2.2. Deep Learning

El Deep Learning es el subconjunto del Machine Learning que se caracteriza por el uso de redes neuronales multicapa llamadas redes neuronales profundas. Al igual que las redes neuronales tradicionales, estas también simulan la capacidad de toma de decisiones del cerebro humano, aunque con más precisión que las anteriores. Estas redes poseen múltiples capas que permiten llevar a cabo aprendizaje no supervisado, donde los datos se procesan progresivamente, generando así predicciones sobre los datos. Este modelo hace que los sistemas que poseen redes neuronales multicapa sean capaces de profundizar en su aprendizaje, identificar patrones complejos y refinar sus resultados de forma continua. Además, son capaces de aprender de sus errores y mejoran continuamente, aumentando y mejorando así la precisión en las tareas que desempeñan.

La siguiente imagen muestra la estructura de multiples capas de las redes reuronales multicapa, así también cómo se relacionan.
Para entender las redes neuronales profundas hay que partir de las redes reuronales tradicionales que se usan en Machine Learning. El Deep Learning amplia y profundiza el MAchine Learning. A diferencia de las redes neuronales tradicionales, que cuentan con una o, como mucho, dos capas, las redes neuronales profundas incluyen varias capas: una capa de entrada, una capa de salida y al menos tres, aunque normalmente cientos, de capas intermedias. Esto se puede ver ilustrado en la Figura 2.2. Además de esta diferencia, existen algunas más importantes que radican en el tipo de datos utilizados, la estructura algorítmica subyacente y los métodos de aprendizaje empleados. En concreto, el Deep Learning elimina la necesidad de intervención humana directa, siendo capaz de comprender y procesar elementos complejos.

El Deep Learning, además de los tipos de aprendizaje de Machine Learning, tam-

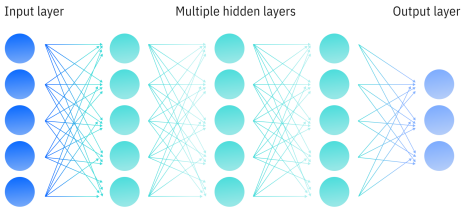

Figura 2.2: Ilustración de las capas de las redes neuronales multicapa y cómo interaccionan.

bién permite implementar otros tipos de aprendizaje como:

- Aprendizaje semisupervisado: este aprendizaje contiene tanto datos etiquetados como y no etiquetados. Se suele utilizar para entrenar modelos de clasificación o regresión.
- Aprendizaje autosupervisado: se encarga de crear etiquetas a partir de datos no estructurados. Así, se evita la necesidad de conjuntos de datos etiquetados manualmente.
- Aprendizaje por transferencia: es capaz de aplicar los conocimientos adquiridos al realizar una tarea o tratar un conjunto de datos para poder mejorar el rendimiento en otra tarea distinta.

2.2.3. IA Generativa

La Inteligencia Artificial generativa parte del Deep Learning, realizando modelos avanzados de Deep Learning que pueden generar contenido original complejo, como texto, imágenes, vídeos o audio. Este contenido se genera a partir de instrucciones o solicitudes realizadas por los usuarios. Estos modelos generativos, que tienen su origen en técnicas estadísticas tradicionales que se usan para el análisis de datos numéricos, aprenden una representación simplificada de los datos que

se usaron para entrenarlos con el fin de poder generar resultados similares a esos datos originales.

El Deep Learning está en auge. En la última década, los modelos han evolucionado significativamente, permitiendo el procesamiento y la generación de datos más complejos. Resultado de esta evolución son la creación de los tres modelos siguientes, que son modelos clave del Deep Learning:

- Los autocodificadores variacionales (VAE): creados en 2013, permiten generar múltiples variaciones del contenido original introducido en la misma instrucción del usuario.
- Los modelos de difusión: introducidos en 2014, se utiliza para crear imagnes nuevas a partir de imagenes que se pasan a los modelos. Funcionan tomando la imagen original y añadiendo ruido hasta que resultan irreconocibles. Después, elimina progresivamente el ruido hasta crear una nueva imágen.
- Los transformadores: son modelos capaces de generar secuencias coherentes de texto, imágenes, vídeo o código a partir de datos secuenciales. Estos transformadores son la base de herramientas como ChatGPT o Copilot.

A continuación, vamos a ver cómo funciona la IA Generativa. Se puede dividir en tres fases:

- Entrenamiento: como todo modelo de IA, partimos del modelo fundacional. En este caso, una red neuronal profunda entrenada con grandes cantidades de datos no estructurados y no etiquetados, como texto, imágenes o vídeos. Este modelo crea una red con miles de millones de parámetros que codifican entidades, patrones y relaciones, permitiendo la generación de contenido autónomo. El entrenamiento de estos modelos usa recursos computacionales muy grandes.
- Ajuste: una vez entrenado, el modelo hay que ajustarlo a las tareas de generación de contenido. Hay dos formas de hacerlo, mediante ajuste fino o fine-tuning, que utiliza ejemplos etiquetados relacionados con la tarea final, y, retroalimentación humana, donde personas reales evalúan las respuestas del modelo.
- Generación y evaluación: después del ajuste, se evalúan regularmente los resultados generados por el modelo. Así, se detectan errores, corrigen desviaciones y mejoran su precisión. Cabe destacar que existe una técnica llamada generación aumentada por recuperación que permite a los modelos acceder a fuentes externas de información en tiempo real lo que mejora la precisión de las respuestas.

2.3. Pasos de un proyecto de Machine Learning

Una vez explica que es la Inteligencia Artificial y sus diferentes ramas, vamos a centrarnos en la rama del Machine Learning, ya que es la que se utiliza para el estudio de este proyecto. En esta sección, se va a explicar los diferentes pasos a realizar dentro de un proyecto de Machine Learning, [20]. Dentro de cada paso, se explicará de forma teórica los diferentes conceptos que se utilicen.

2.3.1. Tratamiento de los Datos

El primer paso corresponde con el estudio y el tratamiento de los datos. Consiste en analizar los datos con los que se va a trabajar y realizarles las transformaciones necesarias para poder utilizarlos para el posterior entrenamiento del modelo. Normalmente, también se realizan tratamientos sobre los datos que, aunque no son necesarios, hacen que los modelos produzcan después mejores resultados y sean más precisos. A continuación, se van a presentar los distintos tratamientos y estudios que se deben hacer sobre el Dataset o conjunto de datos.

Elimiandción de duplicados y procesamiento de tipos

Se trata de eliminar los registros duplicados, ya que no aportan información adicional, sino mayor ruido al modelo. Además, se estudian los distintos tipos de las columnas dentro del Dataset con el objetivo de identificar tipos que no deberían cambiarse o que no son adecuados para el análisis. En cuyo caso, se transforman por los tipos que más se ajusten al tipo original, pudiendo ser utilizados para nuestro estudio del proyecto.

Tratamiento de campos nulos

En Machine Learning, el tratamiento de los campos nulos es fundamental para construir modelos precisos y fiables. Los campos nulos se corresponden con los valores dentro de las categorías que no tienen ningún valor o que faltan. La mayoría de los algoritmos no pueden procesar datos que contengan valores nulos (NaN), lo que implica un procesamiento de estos valores. Esta sección se basa en [21].

Existen varias formas de tratar los valores nulos dependiendo de la cantidad de datos que sean y del tipo. Una solución muy común es rellenar los valores con ceros. Sin embargo, no se recomienda ya que puede distorsionar significativamente los resultados del modelo. Por ello, vamos a ver formas de procesar este tipo de datos que se ajustan a los datos y que no realizan cambios en el Dataset.

Antes de explicar las distintas técnicas de tratamiento de los valores nulos, hay que entender que los datos de un Dataset se dividen en datos numéricos y en datos categóricos. Los datos numéricos son efectivamente los números. Representan información cuantitativa y pueden medirse y contarse. Los datos categóricos son los datos que pueden dividirse mediante la utilización de nombres o etiquetas. Estos datos cualitativos se agrupan en categorías, [22], [23].

1. Eliminar columnas: cuando una columna contiene la mayor parte de sus valores nulos, no aporta información. En estos casos, se elimina la columna, aunque hay que tener cuidado de no perder información valiosa.

2. Eliminar filas: consiste en eliminar las filas que contienen valores nulos. Se realiza cuando solo un pequeño porcentaje del conjunto de datos se ve afectado.

3. Sustitución por la media: ahora vamos a ver técnicas que en lugar de eliminar los datos nulos, los reempalzan por valores nuevos. Un técnica para reañlizar esta sustirución es utilizar la media de la categoría. Se utiliza con datos numéricos cuando no hay valores extremos que pueden distorsionar la media.

4. Sustitución por la moda: reempalzar los valores nulos por la moda de la categoría se usa con variables categóricas o cuando los datos están sesgados. Consiste en sustituir el valor nulo por el valor más frecuente de la columna.

5. Sustitución por la distribución: se reemplazan los campos nulos utilizando la distribución original de la columna. Se utilizan las frecuencias relativas de los datos originales para generar nuevos datos que reemplazarán los valores nulos.

6. Sustitución por categoría faltante: Se utiliza con variables categóticas donde se puede crear una categoría especial que represente explícitamente la falta de datos.

7. Sustitución por un nuevo valor numérico: de la misma manera que la opción anterior pero para datos numéricos, se reemplazan los valores nulos por un valor distintivo como -999 o 0 que indican que el dato no está presente. Hay que tener cuidado del valor que se elige para no confundir al modelo.

En la práctica, en el tratamiento de los valores nulos se utiliza una combinación de las técnicas anteriores. Una opción es utilizar umbrales o porcentajes para la eliminación de las columnas y de las filas, y el resto de datos nulos, sustituirlos por valores nuevos utilizando una o varias de las técnicas presentadas, distinguiendo entre los valores numéricos y los valores categóricos.

Normalización

En esta sección, vamos a distinguir entre los datos numéricos y los datos categóricos.

La normalización de los datos numéricos consiste en transformar los datos de las columnas numéricas para que tengan una escala común. Esta técnica es especialmente útil cuando las variables tienen rangos muy diferentes, lo que puede afectar al rendimiento de algunos algoritmos. La normalización busca unificar las escalas sin distorsionar la distribución original y las relaciones estadísticas originales entre las variables de los datos, [25].

Existen muchos métodos para realizar la normalización de los datos numéricos. Algunos de los más usados son los siguientes:

- Min-Max Scaling: esta técnica escala los datos al rango [0, 1].
- Normalización por percentiles: transforma los datos en percentiles relativos en lugar de usar sus valores absolutos.
- Z-score o estandarización: consiste en centrar los datos con respecto a la media y los escala por su desviación estándar.

Estas técnicas pueden aplicarse a una o varias columnas, siempre y cuando se use la misma para el conjunto entero de datos.

Aunque los modelos solo trabajan con valores numéricos, los conjuntos de datos también suelen contener variables categóricas. Estas deben transformarse a forma numérica para poder utilizarse para el entrenamiento, [26]. A continuación, veamos varias técnicas para realizar esta transformación:

- One-Hot Encoding: esta técnica consiste en convertir cada categoría de la columna otiginal en una columna nueva con datos binarios. Si hay muchas categórias no es muy útil.
- Codificación Ordinal: se sustituye cada categória por un número entero. Esta técnica implica ue existe un orden por lo que solo se debe usar cuando la variable categórica origianl poseea este orden, si no estaría introduciendo un falso orden a la variable.
- Target Encoding: consiste en sustituir cada categoría por la media de la categória que posee el valor objetivo correspondiente. Debe de tenerse cuidado a la hora de aplicarse ya que puede introducir sesgo.

Outliers

Vamos a definir primero qué son los outliers para posteriormente explicar cómo se tratan estos valores. Esta sección se basa en [24].

Los outliers o valores atípicos son puntos de los datos que se desvían considerablemente del resto de puntos del conjunto de datos. Estos puntos desviados pueden tener como causa la variabilidad natural de los datos o deberse a errores de medición o introducción. Es importante estudiar los outliers ya que tienen un gran impacto en el análisis estadístico, puesto que influyen en medidas como la media y la desviación estándar. También, pueden distorsionar la distribución de los datos, lo que afecta a los modelos que utilizan la distribución o asumen la normalidad de los datos.

Los outliers pueden entenderse de dos maneras distintas:

- Univariados: son punto atípicos dentro de una única variable.
- Multivariados: estos puntos son producto del análisis entre dos o más variables. Un valor puede parecer normal por separado, pero resultar atípico en combinación con otros.

A continuación, vamos a ver cómo detectarlos y cómo tratarlos. Para detectar outliers se pueden utilizar varias técnicas. Vamos a presentar algunas de las que más se usan:

1. Histograma: mediante la visuliazación de los datos en un histograma podemos visualizar la distribución de los datos. Si se observan barras muy alejadas de la distribución, indica la presencia de outliers. Es una herramienta útil para una primera inspección ya que no es capaz de señalar exactamente que puntos son los outliers.
2. Diagrama de caja: consiste en un gráfico que muestra la mediana, los cuartiles y posibles outliers. Los valores que se encuentran fuera del rango del diagrama, es decir, que se extienden hasta 1.5 veces del rango intercuartílico, se consideran atípicos.
3. Z-Score: este método mide la desviacion estándar entre un dato respecto a la media. Un dato con una puntuación Z superior a 3, en valor absoluto, indica un outlier. Este método es adecuado para datos que siguen una distribución normal.
4. Z-Score modificado: este método es la adaptación al anterior para dastos que no siguen una distribución normal. Se basa en la mediana y la desviación

absoluta mediana (MAD). En esta técnica se considera un valor atípico el dato cuya puntuación modificada sea superior a 3.5.

5. Rango Intercuartílico(IQR) : consiste en medir la dispersión entre el primer y tercer cuartil. Se consideran los datos outliers si están por debajo del limite inferior o por encima del limite superior. Los límites son los siguientes:

   Límite inferior $= Q1 - 1{,}5 * IQR$

   Límite superior $= Q3 + 1{,}5 * IQR$

Una vez hemos identificado los outliers, utilizando una de las técnicas o una combinación de varias de ellas, tenemos que tratarlos. Tenemos varias formas de hacerlo dependiendo del conjunto de datos:

- Mantenerlos: si los datos se consideran válidos o relevantes para el análisis, se mantienen.
- Eliminarlos: los datos se eliminan cuando hay evidencias de que son producidos por errores de medicion o de entrada.
- Imputarlos: en algunos casos, se pueden sustituir el outlier por la media o la mediana. Esta técnica se usa sobre todo para mantener la integridad del conjunto de datos sin eliminar información.
- Winzorizar: técnica parecida a la anterior salvo que reemplaza los datos por los percentiles más cercanos dentro de un rango estrablecido que suelen serpercentil 5 y 95.

La elección de una de las técnicas o una combinación de ellas, depende del conjunto de datos.

Relación entre las categorías

En Machine Learning, a parte de estudiar las variables, también es importante estudiar cómo se relacionan unas con otras. Por ello, es necesario hacer un estudio de la relación y el impacto que tienen unas columnas sobre el resto del conjunto de datos. Para poder estudiar esta relación, hay que distinguir por el tipo de datos que contengan las columnas, numéricos o categóricos, para aplicar una técnica u otra, [30], [31]. Las técnicas más comunes que se utilizan son las siguientes, distinguiendo en función del tipo de datos:

- Correlación de Pearson: se utiliza cuando las dos columnas contiene datos numéricos. Este coeficiente de correlación nos permite identificar qué variables están relacionadas de forma lineal, lo que nos ayuda a determinar que

variables son útiles para nuestro modelo y cuáles no. La fórmula para calcular este coeficiente es:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

donde r representa el coeficiente de correlación, x_i y y_i son los valores de las dos variables, y $\bar{x}$ y $\bar{y}$ sus respectivas medias. El valor de r está en el intervalo $[-1, 1]$. Cuanto más negativo sea, -1, más inversamente relacionadas están las varibales. Cuando más positivo sea, 1, más positivamente relacionadas están las varibales. Si está cerca del 0, quiere decir que no están muy relacionadas, [32].

- Prueba chi cuadrado: Esta técnica es usada cuando los datos de las dos columnas son de tipo categórico. La prueba consiste en evaluar la independencia de las dos variables categóricas mediante el análisis de la variación de la distribución de una variable a través de los niveles de una segunda variable. Primero crea una tabla de contingencia para cada variables y parte del supuesto de que las dos variables son independientes. Después, calcula las frecuencias esperadas y utilizando el estadístico Chi-cuadrado, compara las frecuencias observadas con las frecuencias esperadas. La fórmula para cálcula la prueba chi cuadrado es:

$$\chi^2 = \sum_{k=1}^{n} \frac{(O_k - E_k)^2}{E_k}$$

donde: O_k es la frecuencia observada para la categoría k, E_k es la frecuencia esperada para la categoría k y n es el número total de categorías, [33], [34].

- Test ANOVA: cuando una de las columnas contiene datos numéricos y la columna restante contiene datos categóricos, se utiliza el test ANOVA para estudiar su relación. ANOVA son las siglas de Análisis de la Varianza. Es una prueba estadística que compara las medias varios grupos, analizando la varianza de cada uno de ellos. La variabilidad entre los grupos indica diferencias en las medias de los grupos. Para determinar esto, se utiliza un estadístico F que cuantifica la relación entre ambas variabilidades. Si el resultado del estadístico es alto, se concluye que al menos una de las medias de grupo es significativamente diferente a las demás, y por tanto, que ese grupo no estría relacionado con el resto. El estadístico F se calcula de la siguiente manera:

$$F = \frac{\frac{\text{SCR}}{k}}{\frac{\text{SCE}}{n-k-1}}$$

donde SCR es la Suma de Cuadrados de la Regresión, SCE es la Suma de Cuadrados de los Errores o residuos, N es número total de observaciones en la muestra y K son los grados de libertad del modelo, [35].

Una vez estudiada la posible relación o no entre las columnas, hay varias formas de tratarlo. En nuestro caso, como se va a realizar una clasificación binaria, se estudia la relación de las columnas con la variable objetivo, o la columna que se quiere predecir. Las columnas relacionadas no se procesan ya que su relación afecta al entrenamiento del modelo. En el caso de que las columnas no estén relacionadas con la variable objetivo, tenemos dos formas de tratarlas:

- Manetenerlas: si las columnas se consideran importantes para el modelo o la legibilidad del mismo, auqnue no afecten al resultado, pueden mantenerse.
- Eliminarlas: lo más recomendable es eliminar las columnas que no esten relacionadas con la variable que se quiere predecir. Esto se debe a que pueden introducir rudio. Además, como los resultados del entrenamiento no se ven afectados no repercute en el modelo final.

Cifrado de datos

Actualmente, es fundamental conservar la integridad, confidencialidad y privacidad. Garantizar la seguridad de los datos se consolida así como un componente esencial para protegerlos de las amenazas.

La seguridad de los datos constituye ya todo un desafío. A medida que las amenazas cibernéticas se vuelven más sofisticadas y persistentes, las diferentes técnicas de cifrado de datos se vuelven obsoletas. Por ello, se requieren soluciones adaptativas, capaces de anticipar, detectar y neutralizar ataques en tiempo real.

Para el cifrado de los datos confidenciales se propone utilizar el algoritmo salted Hash, más concretamente, el método SHA-256. Este algoritmo funciona de la siguiente manera: primero, toma cualquier cadena de caracteres; después, se le introduce el salt, y finalmente, se transforma en una cadena fija de 256 bits en formato hexadecimal. El salt es una cadena de caracteres que se introduce al inicio. Se puede elegir la cadena que se quiera; incluso puede ser aleatoria, y se puede cambiar cuando se requiera.

El algoritmo siempre produce el mismo output con el mismo input. Por ello, para implementar mayor seguridad, se introduce el salt. Proporciona al algoritmo mayor seguridad y robustez frente a ataques. Como el algoritmo produce siempre el mismo output con el mismo input, permite mantener la lógica inicial de

la base de datos. De esta manera, no se alteran los datos, sino que solo se transforman. Además, el output se puede analizar utilizando el operador de igualdad, =.

Es importante tener en cuenta que este proceso es unidireccional. Esto quiere decir que es irreversible a menos que se cree un diccionario. El diccionario incluiría el dato original junto con el dato hasheado para poder volver a obtener el conjunto de datos originales si se desea.

Generación de Datos Sintéticos

Los datos sintéticos son datos no reales, es decir, no generados por humanos, que imitan datos del mundo real. Se generan mediante algoritmos y simulaciones basadas en inteligencia artificial generativa. Lo más importante es que estos conjuntos de datos sintéticos conservan las mismas propiedades que los datos reales que se utilizan para generarlos. Nos basamos en [36] y [37] para realizar esta sección.

Los datos sintéticos son especialmente útiles ya que se pueden generar a demanda y en el volumen que se quiera. Incluso se pueden etiquetar los datos automáticamente, eliminando la necesidad de tratar previamente los datos antes del entrenamiento del modelo. La razón por la que se va a usar en este proyecto es para complementar nuestros conjuntos de datos, proporcionando más cantidad de datos de entrenamiento.

La generación de datos sintéticos hace uso de distintas técnicas para replicar las propiedades de los datos reales en los que se basa. Para generarlos, se pueden utilizar varios métodos:

- Basada en reglas: hace uso de reglas definidas manualmente, basadas en los conocimiento que tenga el usuario del dominio.Garantiza que los datos reflejen las relaciones conocidas, pero puede volverse complejo si hay demasiadas reglas y depende del conocimiento que se tiene.

- Distribución estadística: se basa en la identificación de las distribuciones estadísticas de los datos reales de los que parte y luego generar muestras a partir de esas distribuciones identificadas. Es rápido y fácil de implementar, pero no refleja relaciones complejas.

- Basado en modelos: está técnica consiste en implementar y entrenar un modelo de machine learning con datos reales para aprender sus características. Generamos un modelo capaz de generar datos artificiales que imiten sus propiedades.

- Basada en simulaciones: modela procesos reales mediante simulación de eventos. Es especialemente útil para sistemas dinámicos.Sin embargo, requiere un conocimiento detallado del sistema y suele ser computacionalmente intensivo.

- Modelos generativos: se usan modelos como GAN y VAE que aprenden la distribución de los datos originales y generar nuevas muestras a partir de ellas. Estos modelos permiten generar datos realistas, loq eu los hace especialemente útiles para entrenar modelos de Machine Learning.

Veamos más en profundidad estas tecnologías que se utilizan dentro de los modelos generativos:

- Redes generativas antagónicas (GAN): se generan dos redes neuronales que compiten. Una se encargar de generar datos y otra de evalúarlos. A medida que compiten, el sistema mejora hasta que los datos sintéticos son prácticamente indistinguibles de los reales. Una red genera nuevos datos a partir de los datos originales. La otra red intenta predecir si la salida de datos generada pertenece al conjunto de datos original. Es decir, la red de predicción determina si los datos generados son falsos o reales. El sistema genera nuevos datos hasta que la red de predicción ya no puede distinguir el falso del original, [38].

- Codificadores automáticos variacionales (VAE): consiste en comprimir los datos originales y luego reconstruirlos de forma probabilística.

Separación de los datos

Una vez realizados todos los tratamientos anteriores sobre los datos, el último paso para poder entrenar el modelo con los datos procesados, es dividir el Dataset en dos conjuntos. Esta sección se basa en [27], [28] [29] y [47].

- Conjunto de entrenamiento: se refiere al conjunto de datos que se van a utilizar para el entrenamiento del modelo. Normalemnete, sueel ser el 80 % del conjunto de datos.

- Conjunto de prueba: consiste en el conjunto de los datos que se van a usar para probar el módelo y comrpobar su coprrecto funcionamiento. Sirve como una forma de validación. Se coge el conjunto restante de los datos, es decir, el 20 % restante de los datos.

Cabe destacar que el porcentaje de los conjuntos puede variar y escogerse el que quiera, aunque lo usual es utilizar la mayor parte de los datos para entrenar el modelo y una pequeña porción de los datos para validar su funcionamiento. Este

tipo de división requiere que, después del entrenamiento de los modelos, se use la técnica de crossvalidation para obtener mejores resultados.

Es importante la división del Dataset en estos conjuntos ya que no se debe evaluar un modelo con los mismos datos que se utilizaron para entrenarlo. De esta manera, se obtiene una medición más fiable del rendimiento del modelo.

La validación cruzada es una técnica de Machine Learning que sirve para evaluar la variabilidad del Dataset y la fiabilidad del modelo entrenado. Sobre todo está destinado a detectar si un modelo es sensible a pequeñas variaciones en el conjunto de datos que se usa para su entrenamiento.

La validación cruzada funciona de la siguiente manera:

1. Entrada del Dataset: el conjunto de datos etiquetadose introduce en el modelo.

2. División en pliegues o folds: el conjunto de datos se divide aleatoriamente en varios pliegues o subconjuntos. De forma predeterminada, se utilizan 10 pliegues. Auqnue se puede utilizar cualueir otro número de pliegues.

3. Entrenamiento y validación iterativa: para cada pliegue, se usa como el conjunto de validación, y el resto de pliegues se utilizan para entrenar el modelo. Este proceso se repite hasta que cada pliegue haya sido usado una vez como conjunto de validación.

4. Evaluación: se calculan métricas de rendimiento en cada iteración. También se generan probabilidades y resultados que permiten evaluar la confiabilidad del modelo.

Aunque la validación cruzada suponga un mayor coste computacional añadido al modelo, ofrece múltiples ventajas. La primera ventaja es que utiliza más datos para la evaluación del modelo, ya que todos los datos se van a utilizar en el entrenamiento en algún momento. También permite identificar si el modelo es sensible a variaciones en los datos. Y reduce el riesgo de sobre ajuste ya que introduce en el modelo a diferentes subconjuntos de datos durante la fase de entrenamiento. De esta manera, proporciona una estimación más precisa y robusta. Además de mayor coste computacionales, también requiere múltiples entrenamientos y evaluaciones, lo que puede prolongar el tiempo de procesamiento.

En el conjunto de datos existe otra posible división, que es dividir el Dataset entre los conjuntos de entrenamiento, validación y prueba. Esta forma de dividir

está orientada a la evaluación de los modelos para verificar la precisión de sus predicciones.

Ahora, veamos para qué se utiliza cada conjunto dentro del modelo:

- Conjunto de Entrenamiento o Training Set: tiene cómo objetivo el entrenamiento del modelo. El tamaño que usa habitualmente es el $70\,\% - 80\,\%$ sobre el conjunto total de los datos. El modelo usa este conjunto de datos para aprender los patrones, relaciones y distribuciones que presentan los datos.

- Conjunto de Validación o Validation Set: su objetivo final es ajustar los hiperparámetros y evaluar el rendimiento del proceso de entrenamiento. Normlamente, se utiliza un $10\,\% - 15\,\%$ del total del Dataset. El modelo utiliza este conjunto para tomar decisiones sobre mejoras en el modelo. Para este proceso no utiliza el conjunto de prueba.

- Conjunto de Prueba o Test Set: sirve cuando se termina el proceso para obtener una estimación del rendimiento final del modelo. Se suele usar entre un $10\,\% - 15\,\%$ del total del conjunto de datos. Es importante que este conjunto permanezca siempre separado del resto de conjuntos hasta el final del proceso.

Cabe destacar que en cualquiera de las divisiones presentadas, los porcentajes anteriores deben sumar el $100\,\%$ de los datos. Esto se debe a que no pueden quedar datos sin utilizar.

Hay que tener cuidado con el uso repetitivo de los conjuntos de validación y prueba. Esto se debe a que se "desgastanz se vuelven menos representativos. Utilizar los mismos datos constantemente para ajustar y validar reduce la confianza en los resultados presentados por el modelo. Por ello, se debe periódicamente dividir los datos en nuevos conjuntos de prueba y validación, evitando así este desgaste.

2.3.2. Modelos

A continuación, se van a exponer y explicar en profundidad varios modelos de Machine Learning. Estos modelos son los que se utilizan en el proyecto entrenandolos con los datos correspondientes. Se han elegido los siguientes: Regresión Logística, Random Forest, K-Nearest Neighbors y Red Neuronal Perceptrón Multicapa, ya que se corresponden con un modelo lineal, un modelo basado en árboles, un modelo basado en similitudes y una red neuronal, respectivamente. De esta forma, tenemos cuatro modelos basados en tipos distintos.

Regresión Logistica:

La regresión logística es un modelo estadístico de aprendizaje supervisado ideado para problemas de clasificación binaria y multiclase. En nuestro caso, se emplea en Machine Learning para predecir una variable dependiente binaria. Este apartado se basa en [39], [40].

El modelo utiliza la función sigmoide que transforma los resultados de una ecuación lineal en valores dentro del rango $[0, 1]$. Este valor son las probabilidades finales. Es posible devolver estas probabilidades en vez de la variable binaria. También se utiliza la log-verosimilitud negativa como función de pérdida, y se minimiza mediante descenso del gradiente, lo que equivale estadísticamente a maximizar la log-verosimilitud.

Ahora, veamos en profundidad cómo se comporta el modelo.

La función sigmoide es la siguiente:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

El modelo parte de una combinación lineal de variables independientes. Sin embargo, en lugar de modelar directamente la variable dependiente, modela el logaritmo de la razón de probabilidades, más conocida como odds, como se puede ver a continuación,

$$\log\left(\frac{p(x)}{1 - p(x)}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

Donde la razón de probabilidades es $\frac{p(x)}{1-p(x)}$.

Ahora, despejando $p(x)$ obtenemos,.

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}}$$

Como se puede observar, corresponde con la función sigmoide. Obtenemos así una probabilidad. Si se desea convertir esta probabilidad en una decisión binaria, que es lo más habitual, se define un umbral, por ejemplo, 0.5, y tenemos que la predicción es,

$$y = \begin{cases} 1 & \text{si } p(x) \geq 0{,}5 \\ 0 & \text{si } p(x) < 0{,}5 \end{cases}$$

Los parámetros β se estiman mediante el método de la máxima verosimilitud (MLE). Veamos cómo se calcula. La probabilidad de observar un dato x y su clase

y es,

$$P(y \mid x) = p(x)^y(1 - p(x))^{1-y}$$

Y, la verosimilitud total es,

$$L(\beta) = \prod_{i=1}^{n} p(x_i)^{y_i}(1 - p(x_i))^{1-y_i}$$

Tomando logaritmos obtenemos,

$$\log L(\beta) = \sum_{i=1}^{n} [y_i \log(p(x_i)) + (1 - y_i) \log(1 - p(x_i))]$$

Solo faltaría maximizar esta función mediante gradiente ascendente para obtener los coeficientes óptimos.

Recordemos que los coeficientes están en la escala logarítmica. Por lo tanto, se usa la transformación exponencial para obtener el odds ratio (OR):

$$OR = e^{\beta_j}$$

Random Forest:

El Random Forest o bosque aleatorio es un modelo de ML que se basa en la combinación del resultado de múltiples árboles de decisión con el objetivo de obtener una única predicción más robusta, [41], [42].

Para comprender cómo funciona un bosque aleatorio, primero hay que entender qué es un árbol de decisión. El algoritmo realiza una serie de preguntas binarias, de sí o no, que permiten dividir los datos progresivamente. Cada pregunta se corresponde con un nodo en el árbol y permite seguir ramas distintas según la respuesta sea afirmativa o negativa. El proceso continúa hasta alcanzar un nodo hoja, que contiene la decisión final. Veamos una ilustración de cómo es un árbol de decisión:

Sin embargo, los árboles de decisión individuales tienen una varianza alta y tienden al sobreajuste. Por esta razón, para mitigar estos problemas, el bosque aleatorio crea un conjunto de árboles independientes, es decir, no correlacionados entre sí. Al combinar sus predicciones, conseguimos un modelo más robusto frente a las posibles fluctuaciones del conjunto de datos y proporciona predicciones más estables. El modelo elige la predicción final utilizando el voto mayoritario.

El algoritmo del Random Forest contiene una selección aleatoria de características

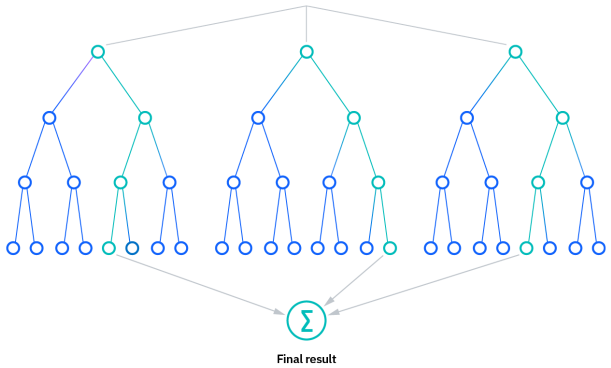

Figura 2.3: Ilustra la estructura binaria en nodos de un árbol de decisión.

en cada división del árbol. Lo que se conoce como "feature bagging". Esto quiere decir que no considera todas las variables para elegir la mejor división, sino que cada árbol evalúa solo un subconjunto aleatorio de características, consiguiendo así reducir más la correlación entre los diversos árboles y mejora la generalización del modelo.

Dentro del algoritmo hay varios hiperparámetros clave para el modelo como el número de árboles a construir, el número de características a muestrear en cada división y el tamaño mínimo de los nodos. Además, cada árbol se entrena sobre una muestra del conjunto de datos de entrenamiento y el resto de estos datos se utilizan para validar el rendimiento del modelo.

En general, el éxito del algoritmo es causa de su capacidad de combinar múltiples modelos individuales, cada uno relativamente débil individualmente, en una única estructura potente y fiable. Este enfoque colaborativo entre modelos permite obtener resultados más precisos, estables y menos propensos al sobreajuste.

K-Nearest Neighbors:

El algoritmo K-nearest neighbors(KNN) o K vecinos más cercanos es un algoritmo de aprendizaje supervisado de Machine Learning que utiliza la proximidad entre los puntos del conjunto de datos para realizar predicciones. Se utiliza para la clasificación binaria partiendo de que los puntos similares suelen encontrarse cerca unos de otros en el espacio de características de los datos, [43], [44].

Veamos en profundidad cómo funciona el modelo. El algoritmo asigna una etiqueta a cada punto del conjunto, basándose en una votación de mayoría entre sus puntos vecinos más cercanos. Esta técnica se conoce como votación por pluralidad.

Cabe destacar que no es necesario que una clase tenga más del 50 % de los votos; sino que basta con que tenga más votos que los demás. Veamos con una ilustración cómo se asignan a las etiquetas a los nuevos puntos a partir de sus puntos vecinos:

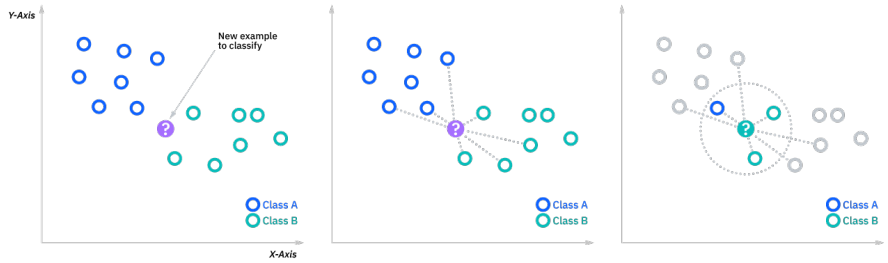

Figura 2.4: Ilustra cómo depende la etiqueta de cada punto de sus punto vecinos.

Para aplicar el modelo, es necesario establecer sus dos hiperparámetros fundamentales: la métrica de distancia y el valor de k. La distancia determina la cercanía entre los puntos de datos. A continuación, vamos a introducir las más populares:

1. Distancia euclidiana: calcula la línea recta entre los dos puntos en un espacio de características continuas. Se calcula con la siguiente fórmula:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}.$$

2. Distancia Manhattan: mide la suma de las diferencias absolutas. Es esialemente útil para estructuras en cuadrícula. Se calcula como sigue:

$$d(x, y) = \left(\sum_{i=1}^{m} |x_i - y_i|\right).$$

3. Distancia de Minkowski: generaliza las distancias anteriores introduciendo un parámetro p. La fórmula es la siguiente:

$$d(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}.$$

4. Distancia de Hamming: mide la cantidad de posiciones diferentes entre dos vectores de igual longitud. Se calcula con la siguiente fórmula:

$$D_H = \left(\sum_{i=1}^{k} |x_i - y_i| \right).$$

El otro hiperparámetro, valor de k, indica cuántos vecinos se considerarán para tomar la decisión. Elegir el valor correcto de k es crucial. Un valor bajo puede llevar al sobreajuste, mientras que un valor alto puede causar subajuste. Se recomienda utilizar técnicas como la validación cruzada para encontrar el k óptimo, así como escoger un número impar para evitar empates..

A continuación, se va a describir las distintas etapas que tiene el algoritmo:

- Etapa 1: selecciona los hiperparámetros: el número de K vecinos y la distancia.
- Etapa 2: calcular la distancia, siempre utilizano la misma.
- Etapa 3: toma los K puntos vecinos más cercanos según la distancia calculada.
- Etapa 4: entre los K puntos vecinos, cuenta el número de puntos en cada categoría.
- Etapa 5: atribuye el nuevo punto a la categoría más presente entre los K puntos vecinos.

El algoritmo KNN se considera del tipo .ªprendizaje perezoso". Esto se debe a que no realiza un entrenamiento explícito de los datos. En lugar de aprender un modelo, simplemente almacena los datos de entrenamiento y realiza cálculos sobre ellos para determinar la predicción. Esta característica le permite adaptarse fácilmente a nuevos datos.

Además, KNN destaca por tener pocos hiperparámetros y se adapta a nuevos datos sin necesidad de reentrenamiento. No obstante, haya que tener en cuenta que no escala bien grandes volúmenes de datos, lo que puede resultar en altos costos computacionales. También es susceptible a que su rendimiento disminuya cuando el número de características crece. También es propenso al sobreajuste, recomendándose utilizar la técnica de validación cruzada.

Red Neuronal Perceptrón Multicapa:

Un perceptrón multicapa(MLP) es un tipo de red neuronal formada por neuronas totalmente conectadas que utilizan una función de activación no lineal. Se componen de una capa de entrada, varias capas ocultas y una capa de salida. Cabe destacar que en realidad estas redes neuronales están formadas por neuronas sigmoidales, no por perceptrones, ya que en la vida real, los problemas suelen ser no lineales. Por lo tanto, su uso está enfocado para distinguir datos que no son linealmente separables. Para esta sección se utiliza [45] y [46].

Veamos más en detalle las distintas capas que componen un MLP:

1. La capa de entrada: está formada por neuronas que reciben los datos de entrada. Cada neurona representa una característica de los datos. El número de neuronas de la capa de entrada viene dado por lo tanto por el número de características del conjunto de los datos de entrada.

2. La capa oculta: se corresponde con las capas intermedias entre la capa de entrada y de salida y suelen ser varias capas de neuronas. Cada neurona de una capa oculta recibe entradas de todas las neuronas de la capa anterior, puede ser tanto la capa de entrada como otra capa oculta, y produce una salida que pasa a la capa siguiente. El número de capas ocultas y el número de neuronas de cada capa oculta son hiperparámetros.

3. La capa de salida: esta es la capa donde las neuronas producen la salida final de la red. El número de neuronas de esta capa depende del conjunto de datos y del objetivo del modelo. En clasificación binaria, suele haber una o dos neuronas, auqnue dependiendo de la función de activación, y representan la probabilidad de pertenecer a una clase un otra.

Las neuronas de capas contiguas están totalmente conectadas entre sí. A cada conexión se le asigna un peso durante la fase de entrenamiento, que determina la fuerza de la conexión. Además, cada capa, menos la capa de entrada, incluye una neurona de polarización o de sesgo. Esta se encarga de proporcionar una entrada constante a las neuronas de la capa siguiente. Las neuronas de sesgo tienen su propio peso asociado a cada conexión. Esta neurona de sesgo comparte la función de activación a la capa siguiente. Ajustando los pesos que contiene la neurona de polarización, el algoritmo puede ajustar el umbral de activación y ajustarse mejor a los datos del conjunto de entrenamiento. Veamos esta estructura gráficamente:

Como se ha mencionado en el párrafo anterior, cada neurona de las capas ocultas y de la capa de salida utiliza una función de activación que calcula su suma ponderada de entradas. Funciones de activación habituales son sigmoide, tanh, ReLU y

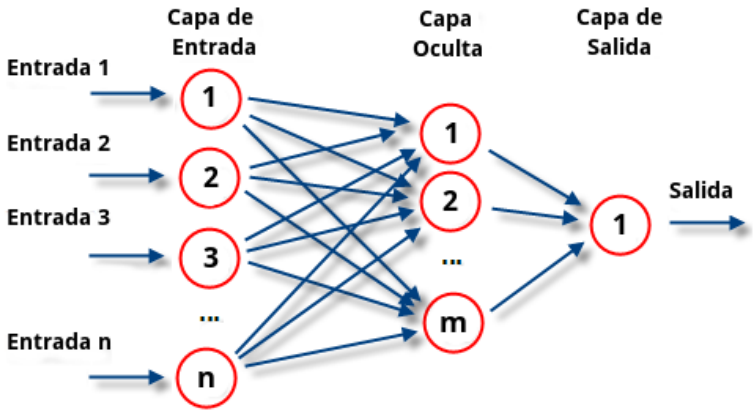

Figura 2.5: Ilustra la estructura de capas de la red neuronal MLP y sus pesos en cada conexión.

softmax. Estas funciones son las que introducen la no linealidad en la red neuronal.

Para entrenar las redes MLP, se realiza mediante el algoritmo de retropropagación, que calcula los gradientes de una función de pérdida con respecto a los parámetros del modelo y actualiza los parámetros iterativamente para minimizar la pérdida. A continuación, veamos más en detalle este funcionamiento de un perceptrón multicapa, capa a capa:

1. Primero, la capa de entrada de un MLP recibe datos de entrada del conjunto de entrenamiento. Cada neurona de la capa de entrada representa una característica de los datos. En esta capa, las neuronas no realizan ningún cálculo, simplemente trasnmiten los valores de entrada a las neuronas de la primera capa oculta.

2. Una vez se ha trasnmitido la infromación de las neuronas de la capa de entrada a la primera capa oculta en la que cada neurona de la capa oculta recibe información de todas las neuronas de la capa anterior, la infromación de entrada se multiplican por los pesos correspondientes, denotados como w. Recordemos que las capas ocultas están formadas por neuronas interconectadas. Además de los pesos, cada neurona tiene un sesgo asociado, denotado como b. El sesgo proporciona un parámetro adicional que permite ajustar el umbral de salida de cada neurona. Ambos parámtros, se aprenden durante el entrenamiento. Cada neurona calcula la suma ponderada de sus entradas de la siguiente manera,

$$z = \sum_{i=1}^{n} w_i x_i + b,$$

donde n es el número total de conexiones de entrada, w_i es el peso de la

entrada i-ésima y x_i es el valor de la entrada i-ésima. Después, esta suma se pasa por la función de activación $f(z)$ elegida y este valor se pasa a la siguiente capa correspondiente.

3. Finalemente, la capa de salida produce las predicciones finales de la red neuronal. En esta capa, cada neurona recibe la entrada de las neuronas de la capa oculta contigua y aplica la función de activación escogida. Esta función de activación suele ser distinta a la usada en las capas ocultas. Durante el entrenamiento, la red ajusta los pesos para minimizar la diferencia entre las salidas previstas y los valores reales. Esto se logra utilizando un algoritmo de optimización como el descenso de gradiente estocástico (SGD).

Veamos el algoritmo de descenso de gradiente estocástico en detalle. El SGD comienza con el conjunto de parámetros de pesos y sesgos del modelo. Después, aplica la optimización iterativa. Consiste en encontrar el mínimo de una función de pérdida. Para ello, el algoritmo se mueve iterativamente en la dirección de la disminución más pronunciada del valor de la función. Para cada iteración se realizan una serie de pasos:

1. Baraja los datos de entrenamiento.
2. Divide los datos en minilotes.
3. Para cada minilote,
   - Calcula el gradiente de la función de pérdida respecto a los parámetros del modelo.
   - Después, actualiza los parámetros del modelo en la dirección opuesta al gradiente, utilizando una tasa de aprendizaje:

     $$\theta_{t+1} = \theta_t - \eta \nabla J(\theta_t),$$

     donde θ_t representa los parámetros en la ietración t y $\nabla J(\theta_t)$ es el gradiente de la función de périda J con respecto a los parámetros.

El tamaño del paso que se da en cada iteración del descenso gradiente viene dado por un parámetro llamado tasa de aprendizaje, antes llamado η. Este parámetro indica el tamaño de los pasos que dan hacia el mínimo en cada iteración. Este proceso se repite un número máximo de iteraciones o hasta que se cumpla un criterio de convergencia, como que el cambio en la función de pérdida esté por debajo de un determinado umbral. Cabe destacar que el SGD es computacionalmente eficiente y puede tratar grandes conjuntos de datos.

Para finalizar, vamos a ver la retropropagación. Es un método que se utiliza para actualizar los pesos de la red usando los gradientes calculados por cada minilote. Funciona de la siguiente manera:

1. Pase hacia delante: primero, se introducen los datos y se calcula la salida capa por capa.
2. Cálculo de pérdidas: aquí, se calcula la diferencia entre la salida de la red y el valor objetivo.
3. Paso hacia atrás: ahora, se calculan los gradientes usando la regla de la cadena.
4. Actualización de parámetros: finalemente, se ajustan los pesos usando SGD u otro optimizador.

Este proceso se repite múltiples iteraciones hasta que el modelo converge y minimiza la función de pérdida, mejorando así su predicción.

2.3.3. Afinamiento de los modelos

Una vez que los modelos seleccionados han sido entrenados utilizando el conjunto de datos de entrenamiento, se evaluan y analizan sus resultados. Este análisis incluye la medición de diferentes métricas, como la evaluación de su precisión utilizando el conjunto de datos de prueba, análisis del rendimiento de cada modelo, validación cruzada y el ajuste de los hiperparámetros de cada modelo.

Tras calcular estas métricas de rendimiento para cada modelo, se determinan las posibles mejoras y se aplican con el objetivo de mejorar la capacidad predictiva del modelo. A continuación, se describen algunas de las técnicas y métodos más comunes que se emplean en Machine Learning.

Exactitud y Precisión

Existitud o accuracy es una medida del rendimiento de un modelo de Machine Learning que mide la proporción de predicciones que han sido correctamente identificadas por el modelo. Se evalua tomando el conjunto de prueba e introduciéndolo en el modelo. Después, se comparan los resultados de la predicción con los datos originales. Nos basamos en [51] y [50].

La precisión en cambio, mide la capacidad del modelo para evitar falsos positivos. Se calcula tomando las predicciones positivas correctas y diviéndolas entre la suma entre las predicciones positivas correctas y las predicciones positivas incorrectas.

Matriz de Confusión

La matriz de confusión expone de forma gráfica cómo el modelo clasifica correctamente o incorrectamente las distintas clases predichas por el modelo. Cada celda de la matriz muestra el número de muestras que pertenecen a la clase real y cuál fue el resultado de su predicción por parte del modelo. De esta manera, podemos ver las clases originales de los datos y en qué clases ha predicho el modelo que pertenecen. veamos un ejemplo ilustrativo, [50]. Como podemos observar en

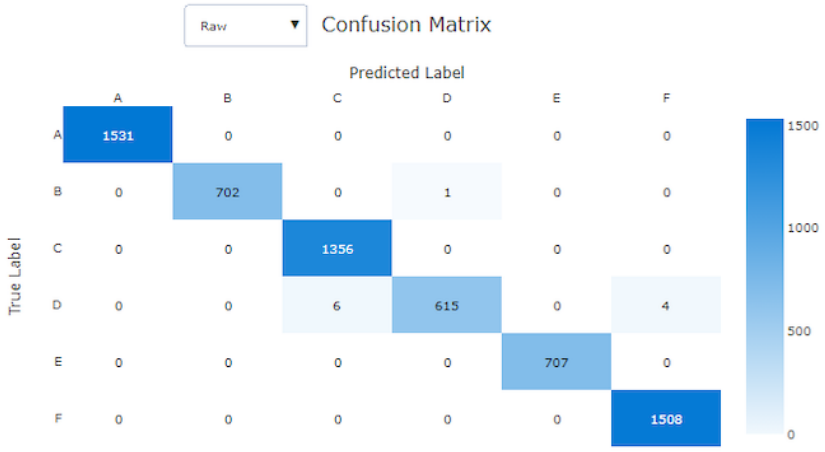

Figura 2.6: Muestra un ejemplo de una matriz de confusión.

la Figura 2.5, las celdas de la diagonal coinciden con las clases que el modelo ha predicho correctamente. El resto de celdas muestran las clases que el modelo ha predicho incorrectamente. Por lo tanto, un buen modelo tiene la mayoría de los datos en la diagonal de la matriz.

Curva ROC

La curva ROC es otra representación que ilustra gráficamente la tasa de verdaderos positivos frente a la de falsos positivos. La AUC (área bajo la curva) evalua la capacidad del modelo para distinguir entre clases. Se puede interpretar como la proporción de las muestras clasificadas correctamente, más concretamente, es la probabilidad de que el modelo clasifique una muestra positiva más arriba que una muestra negativa elegidas aleatoriamente. Un AUC próximo a 1 indica un modelo excelente, mientras que próximo a 0.5 refleja un modelo aleatorio. La forma de la curva también revela si hay un desequilibrio significativo entre clases, [50].

Vamos a ilustrar estos conceptos en dos imágenes. La primera, muestra la representación de la curva ROC de un modelo óptimo. La segunda, muestra la curva ROC de un modelo con una mala predicción de las clases.

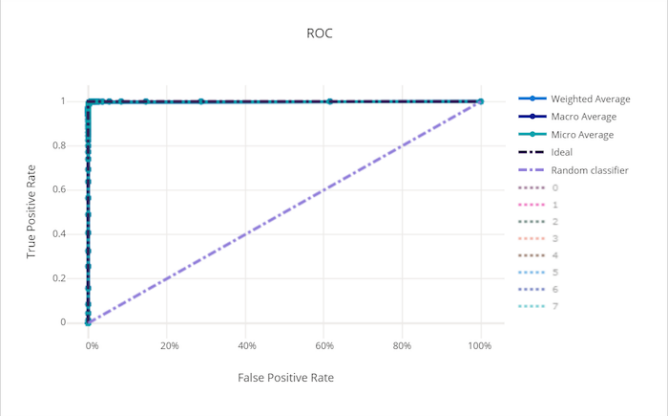

Figura 2.7: Representación de la curva ROC de un modelo que ha predicho correctamente las clases del conjunto de datos.

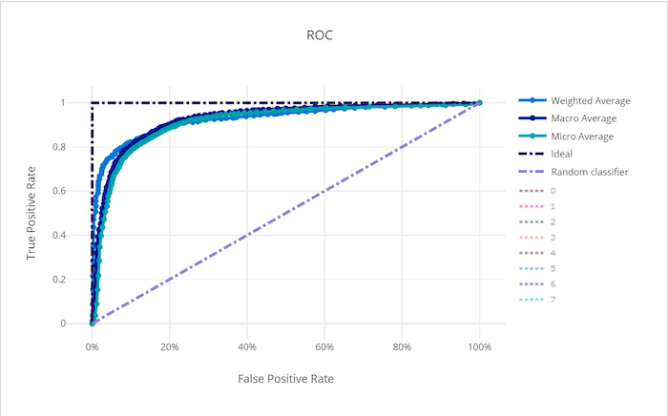

Figura 2.8: Representación de la curva ROC de un modelo que ha predicho incorrectamente las clases del conjunto de datos.

Validación Cruzada

La validación cruzada o cross-validation es un método para evaluar el rendimiento de un modelo predictivo de Machine Learning. Esta técnica permite conocer la eficacia del modelo, [47].

Se emplea después de entrenar un modelo con datos etiquetados para garantizar la exactitud de las predicciones obtenidas. Para lograr esto, es necesario validar el modelo, que consiste en determinar si el modelo es suficientemente fiable para representar los datos o no.

Evaluar este rendimiento de un modelo implica probarlo con datos nuevos y desconocidos. La validación cruzada considera un procedimiento de re-sampling o remuestreo, sin necesidad de obtener nuevos datos sino utlizando los datos originales. Para llevarlo a cabo, es necesario reservar antes del entrenamiento una parte del conjunto de datos de entrenamiento. Esta porción de datos no se usará para entrenar el modelo, sino para probarlo y validar su desempeño posteriormente.

A continuación, se presentan dos técnicas de validación cruzada:

- La técnica del Train-Test Split: como se menciono en el tratamiento de los datos, este método consiste en dividir el conjunto de datos en dos partes: una para entrenar el modelo y otra para validar su rendimiento. Generalmente, se utlizando unos porcentajes de entre un 70 % y un 80 % de los datos para el entrenamiento y los 20 % o 30 % restantes se destina a la validación mediante cross-validation.

- La técnica K-Folds: a diferencia del Train-Test Split, garantiza que todos los datos del conjunto original formen parte tanto del conjunto de entrenamiento como el de prueba. El proceso comienza dividiendo aleatoriamente el conjunto de datos en K grupos o folds. El valor de K no debe ser ni demasiado pequeño ni demasiado grande. Normlamente, se elige un número entre 5 y 10. Un valor alto de K reduce el sesgo del modelo, pero puede aumentar la varianza y generar sobreajuste. Ahora, si se escoge un valor pequeño, se comporta muy parecido al método Train-Test Split. K-folds ajusta el modelo usando K-1 folds, y luego lo valida con el fold que falta. En cada fold, se fuardan las métricas de desempeño y los errores, y este proceso se repite hasta que cada fold haya sido utilizado como conjunto de prueba. Finalmente, se calcula la media de las puntuaciones obtenidas para obtener la métrica final del rendimiento del modelo.

Ajuste de los hiperparámeros

El ajuste de hiperparámetros es el proceso que identifica y selecciona los valores óptimos de los hiperparámetros de cada modelo. Cada modelo y su conjunto de datos correspondiente requiere un conjunto específico de hiperparámetros. A diferencia de los parámetros internos del modelo, que se ajustan automáticamente durante el entrenamiento, los hiperparámetros deben establecerse manualmente. Para esta sección se ha usado [48] y [49].

Veamos primero cómo se definen los hiperparámetros. Los hiperparámetros son variables externas al modelo definidas antes del entrenamiento que influyen en el proceso de entrenamiento del modelo. Los hiperparámetros influyen directamente en la estructura, las funciones y el rendimiento del modelo. Por lo tanto, es necesario una buena elección para obtener buenos resultados. Una mala configuración puede llevar al subajuste o sobreajuste del modelo. El ajuste correcto permite un equilibrio entre velocidad de entrenamiento y precisión, lo que produce modelos más precisos, eficientes y robustos.

El proceso de ajuste consiste en probar diferentes combinaciones de hiperparámetros y evaluar su rendimiento. El objetivo es encontrar la configuración óptima que minimice la función de pérdida y maximice la precisión del modelo. Esto ayuda a encontrar el mejor equilibrio entre sesgo (precisión) y varianza (consistencia en nuevos datos). Este proceso es iterativo. Primero, se definen las métricas objetivo como la precisión, después se prueban distintas combinaciones de hiperparámetros y se analiza el rendimiento del modelo. Este proceso de ajuste de hiperparámetros requiere altos recursos computacionales.

El proceso de ajuste puede ser realizado manualmente o automáticamente. El ajuste manual es lento y tedioso, pero permite comprender cómo afectan los diferentes hiperparámetros al modelo. A día de hoy, se utilizan técnicas automáticas que permiten probar múltiples combinaciones.Veamos a continuación varios métodos automáticos para el ajuste de los hiperparámetros:

- Optimización bayesiana: parte del teorema de Bayes para construir un modelo probabilístico utilizando los hiperparámetros. En cada iteración, usa los resultados de las anteriores para seleccionar la próxima combinación, mejorando progresivamente los resultados.
- Búsqueda por cuadrícula o grid search: consiste en definir un conjunto de valores posibles para cada hiperparámetro y probar todas las combinaciones. Es una técnica exhaustiva que genera buenos resultados, pero muy costosa computacionalmente.

- Búsqueda aleatoria: en lugar de probar todas las combinaciones, selecciona aleatoriamente combinaciones de hiperparámteros dentro de un rango.
- Hiperbanda: es una técnica que parte de la busqueda aleaoria y la mejora quitando el uso de configuraciones de hiperparámetros que no ofrecen resultados sólidos y asignando más recursos a configuraciones prometedoras.

Ahora, veamos los hiperparámetros que se pueden ajustar dentro de los modelos presentados en la sección anterior:

- Regresión Logística: algunos de los hiperparámetros que se pueden encontrar en el modelo son C, que representa el inverso de la fuerza de regularización; penalty, que define el tipo de regularización a aplicar; solver, que determina el algoritmo de optimización; y maxiter, indicando el número máximo de iteraciones para que el algoritmo de optimización converja.
- Random Forest: el parámetro más conocido es nestimators, que indica el número de árboles que se construirán. Maxdepth controla la profundidad máxima de los árboles. Minsamplessplit indica el número mínimo de muestras necesarias para dividir un nodo, y minsamplesleaf, el mínimo de muestras requeridas en una hoja. El parámetro maxfeatures define cuántas características se consideran al buscar la mejor división. También se tiene bootstrap, que indica si se usará muestreo con reemplazo, y criterion, que especifica la función de evaluación de la calidad de cada división.
- K-Nearest Neighbors: los hiperparámetros fundamentales son nneighbors, que determina cuántos vecinos se consideran para clasificar una muestra, weights que define cómo se ponderan los vecinos, y la elección de la distancia utilizadosela distancia de 'minkowski' por defecto.
- Perceptrón Multicapa: este modelo incluye varios hiperparámetros como hiddenlayersizes, que indica el número y tamaño de las capas ocultas, activation que corresponde a la función de activación, solver que se refiere al algoritmo de optimización y el parámetro alpha, que se emplea para controlar la regulación y actúa como coeficiente de penalización. Además, existen también learningrate para refinar el aprendizaje del modelo y que define cómo cambia la tasa de aprendizaje durante el entrenamiento, el valor inicial de esa tasa que se fija con learningrateinit, y maxiter que controla el número máximo de iteraciones del entrenamiento. Además, earlystopping permite detener el entrenamiento si el rendimiento en el conjunto de validación no mejora.

Capítulo 3

Desarrollo y Evaluación

En este capítulo, se va a implementar y realizar el estudio del proyecto, así como después, se van a evaluar los resultados que se obtengan. Los pasos que se van a seguir son los descritos en el capítulo anterior. Sin embargo, antes de comenzar con el tratamiento de los datos, vamos a describir los dos conjuntos o Datasets a partir de los cuales se van a entrenar los modelos.

3.1. Descripción de los conjuntos de datos

Se dispone de dos conjuntos de datos distintos sobre incendios forestales para el estudio del proyecto. Ambos poseen información sobre las características y las métricas de incendios forestales ocurridos. La mayoría de estas características se corresponden con las características ambientales que se tenían en el momento en que comenzaron los incendios. Los dos conjuntos han sido tomados del repositorio de datos del que dispone la universidad californiana UC Irvine, UC Irvine Machine Learning Repository, [52].

- Dataset 'forestfires': Este Dataset contiene información sobre incendios forestales ocurridos en Portugal. Más concretamiento, contiene incendios forestales ocurridos en el parque Montesinho, en el norte del país, en 2007. En total son 518 registros o incendios.

- Dataset 'Algerian_forest_fires_dataset': Este Dataset contiene información sobre incendios forestales ocurridos dos bosques argelinos. Contiene información sobre incendios ocurridos en la región de Bejaia en el noreste de Argelia y en la región de Sidi Bel-abbes situada en el noroeste de Argelia. Contiene registros sobre 244 incendios ocurridos durante 2012.

A continuación, vamos a describir detalladamente las características de cada uno. Antes de continuar, conviene entender el sistema de índices FWI ya que ambos

conjuntos utilizan estas métricas en sus características. El Índice de Tiempo de Incendios o Fire Weather Index (FWI), es un índice que se basa en la meteorología para estimar el peligro de incendio. Fue desarrollado por el Servicio Forestal Canadiense para estimar las condiciones de ignición y propagación de incendios forestales en base a varias variables climáticas como temperatura, precipitación, humedad relativa y velocidad del viento. Los índices concretos se explicarán más adelante. La información referente al sistema FWI se basa en [53], [54], [55], [56].

Empecemos por el conjunto de datos de los incendios ocurridos en Portugal. Cada incendio que contiene el conjunto de datos tiene las siguientes características:

- X: de tipo entero, se correspode con la coordenada X espacial.
- Y: de tipo entero, se correspode con la coordenada Y espacial.
- Month: representa el mes del año. Viene dado con las tres primeras letras del mes correspondiente en inglés. Por lo tanto, es de tipo categórico.
- Day: esta característica representa el día de la semana. Viene dado con las tres primeras letras del día correspondiente en inglés. Por lo tanto, es de tipo categórico.
- FFMC: Fine Fuel Moisture Code (FFMC) es una de las métricas del índice FWI. Estima el contenido de humedad de los combustibles finos muertos y de la materia orgánica presente en la capa superficial del suelo de aproximadamente 1.2 cm de profundidad. El FFMC es un valor numérico que varía en el rango de 18.7 a 96.20 en este Dataset. Este índice es un buen indicador de la probabilidad de ignición de incendios provocados por chispas o partículas incandescentes, ya que los combustibles finos son muy susceptibles de inflamarse.
- DMC: Duff Moisture Code o Código del Contenido de Humedad de la Hojarasca mide la humedad del mantillo y se corresponde con el contenido de humedad de los combustibles de tamaño mediano, así como de la materia orgánica presente en una capa de suelo de aproximadamente 7 cm de profundidad. Su valor númerico varía entre 1.1 y 291.3. Este índice predice cómo arden los combustibles situados en el estrato medio del mantillo.
- DC: Drought Code o Código de Sequía mide la humedad de los estratos inferiores del mantillo. Estima el contenido de humedad de los combustibles de gran tamaño y de la materia orgánica presente en una capa de suelo de unos 18 cm de profundidad. Su valor numérico oscila entre 7.9 y 860.6. Se considera un buen indicador de los efectos estacionales de la sequía en los combustibles de gran tamaño.

- ISI: Initial Spread Index o Índice de Propagación Inicial evalúa el riesgo de propagación de un incendio forestal. Calcula su peligrosidad combinando el estado de los combustibles finos muertos (FFMC) con la velocidad del viento. Su valor, que oscila entre 0.0 y 56.10, estima la velocidad de propagación del fuego en el frente de llamas sin medidas de extinción y sin tener en cuenta la variabilidad del combustible.

- temp: esta característica de tipo numérico, se refiere a la temperatura ambiente medida en grados centígrados. En este caso, oscila entre 2.2 to 33.30 grados.

- RH: Mide el porcentaje de humedad ambiente. Los valores numéricvos van desde el 15 % hasta el 100 %.

- wind: Se refiere a la velocidad del viente. Se mide en km/h y sus valores se encuentran entre 0.40 y 9.40.

- rain: Se refiere a la lluvia. Se mide en mm/m2 y sus valores oscilan entre 0.0 hasta 6.4.

- area: mide el area quemada del incendio que se ha provocado. Se mide en hectareas y oscila entre 0.00 y 1090.84. Cabe destacar que un area quemada de 0.0 implica que ha habido un incendio pero que no se ha propagado.

Veamos una muestra del Dataset: Continuemos con el conjunto de datos de los

```
X,Y,month,day,FFMC,DMC,DC,ISI,temp,RH,wind,rain,area
7,5,mar,fri,86.2,26.2,94.3,5.1,8.2,51,6.7,0,0
7,4,oct,tue,90.6,35.4,669.1,6.7,18,33,0.9,0,0
7,4,oct,sat,90.6,43.7,686.9,6.7,14.6,33,1.3,0,0
8,6,mar,fri,91.7,33.3,77.5,9,8.3,97,4,0.2,0
8,6,mar,sun,89.3,51.3,102.2,9.6,11.4,99,1.8,0,0
8,6,aug,sun,92.3,85.3,488,14.7,22.2,29,5.4,0,0
8,6,aug,mon,92.3,88.9,495.6,8.5,24.1,27,3.1,0,0
8,6,aug,mon,91.5,145.4,608.2,10.7,8,86,2.2,0,0
8,6,sep,tue,91,129.5,692.6,7,13.1,63,5.4,0,0
7,5,sep,sat,92.5,88,698.6,7.1,22.8,40,4,0,0
7,5,sep,sat,92.5,88,698.6,7.1,17.8,51,7.2,0,0
7,5,sep,sat,92.8,73.2,713,22.6,19.3,38,4,0,0
6,5,aug,fri,63.5,70.8,665.3,0.8,17,72,6.7,0,0
6,5,sep,mon,90.9,126.5,686.5,7,21.3,42,2.2,0,0
6,5,sep,wed,92.9,133.3,699.6,9.2,26.4,21,4.5,0,0
```

Figura 3.1: Se muestran los primeros 15 registros de los incendios forestales en Dataset 'forestfires'.

incendios ocurridos en Argelia. Cada incendio que contiene el conjunto de datos tiene las siguientes características:

- region: se correspode con la región argelina. Es de tipo categórico y puede ser Bejaia o Sidi-Bel Abbes.
- day: esta característica representa el día del mes. Viene dado el número del día correspondiente. Por lo tanto, es de tipo numérico.
- month: representa el mes del año. Viene dado por el número del mes correspondiente. Por lo tanto, es de tipo numérico.
- year: representa el año. Viene dado por el número del año correspondiente. Por lo tanto, es de tipo numérico. En este caso, siempre es el año 2012.
- FFMC: Fine Fuel Moisture Code (FFMC) es una de las métricas del índice FWI. Estima el contenido de humedad de los combustibles finos muertos y de la materia orgánica presente en la capa superficial del suelo de aproximadamente 1.2 cm de profundidad. El FFMC es un valor numérico que varía en el rango de 28.6 a 96.200 en este Dataset. Este índice es un buen indicador de la probabilidad de ignición de incendios provocados por chispas o partículas incandescentes, ya que los combustibles finos son muy susceptibles de inflamarse.
- DMC: Duff Moisture Code o Código del Contenido de Humedad de la Hojarasca mide la humedad del mantillo y se corresponde con el contenido de humedad de los combustibles de tamaño mediano, así como de la materia orgánica presente en una capa de suelo de aproximadamente 7 cm de profundidad. Su valor númerico varía entre 0.7 y 65.9. Este índice predice cómo arden los combustibles situados en el estrato medio del mantillo.
- DC: Drought Code o Código de Sequía mide la humedad de los estratos inferiores del mantillo. Estima el contenido de humedad de los combustibles de gran tamaño y de la materia orgánica presente en una capa de suelo de unos 18 cm de profundidad. Su valor numérico oscila entre 6.9 y 220.4. Se considera un buen indicador de los efectos estacionales de la sequía en los combustibles de gran tamaño.
- ISI: Initial Spread Index o Índice de Propagación Inicial evalúa el riesgo de propagación de un incendio forestal. Calcula su peligrosidad combinando el estado de los combustibles finos muertos (FFMC) con la velocidad del viento. Su valor, que oscila entre 0.0 y 19.00, estima la velocidad de propagación del fuego en el frente de llamas sin medidas de extinción y sin tener en cuenta la variabilidad del combustible.

- Temperature: esta característica de tipo numérico, se refiere a la temperatura ambiente medida en grados centígrados. En este caso, oscila entre 2.2 to 33.30 grados.
- BUI: Build-Up Index o Índice de Combustible Disponible estima el combustible total disponible, tanto de partículas medias como gruesas, para la combustión y propagación del fuego, incluyendo los combustibles pesados que se hallan en el suelo y que pueden alimentar el fuego. Es de tipo numérico y oscila entre 1.1 y 68.00.
- FWI: Fire Weather Index o Índice Meteorológico de Incendios Forestales constituye una medida de la probabilidad de ignición. Está relacionada con el contenido de humedad de los combustibles, la posible extensión del incendio y con la dificultad de su extinción. El índice representa la intensidad de propagación del fuego, medida como energía por unidad de longitud del frente del incendio. Se suele utilizar como un indicador de comportamiento del fuego. Es de tipo numérico y oscila entre 0.0 y 31.10.
- RH: Mide el porcentaje de humedad ambiente. Los valores numéricos van desde el 21 % hasta el 90 %.
- Ws: Se refiere a la velocidad del viente. Se mide en km/h y sus valores se encuentran entre 6.0 y 29.00.
- Rain: Se refiere a la lluvia. Se mide en mm/m2 y sus valores 0scilan entre 0.0 hasta 16.80.
- Classes: Esta variable de tipo categórico indica si en las condiciones ambientales antes descritas ha habido un incendio o no. Tiene dos valores 'fire', indicando que si se ha provocado un incendio forestal, y 'not fire', indicando que no se ha provocado ningún incendio.

Veamos una muestra del Dataset:

```
day,month,year,Temperature, RH, Ws,Rain ,FFMC,DMC,DC,ISI,BUI,FWI,Classes
01,06,2012,29,57,18,0,65.7,3.4,7.6,1.3,3.4,0.5,not fire
02,06,2012,29,61,13,1.3,64.4,4.1,7.6,1,3.9,0.4,not fire
03,06,2012,26,82,22,13.1,47.1,2.5,7.1,0.3,2.7,0.1,not fire
04,06,2012,25,89,13,2.5,28.6,1.3,6.9,0,1.7,0,not fire
05,06,2012,27,77,16,0,64.8,3,14.2,1.2,3.9,0.5,not fire
06,06,2012,31,67,14,0,82.6,5.8,22.2,3.1,7,2.5,fire
07,06,2012,33,54,13,0,88.2,9.9,30.5,6.4,10.9,7.2,fire
08,06,2012,30,73,15,0,86.6,12.1,38.3,5.6,13.5,7.1,fire
09,06,2012,25,88,13,0.2,52.9,7.9,38.8,0.4,10.5,0.3,not fire
10,06,2012,28,79,12,0,73.2,9.5,46.3,1.3,12.6,0.9,not fire
11,06,2012,31,65,14,0,84.5,12.5,54.3,4,15.8,5.6,fire
12,06,2012,26,81,19,0,84,13.8,61.4,4.8,17.7,7.1 ,fire
13,06,2012,27,84,21,1.2,50,6.7,17,0.5,6.7,0.2,not fire
14,06,2012,30,78,20,0.5,59,4.6,7.8,1,4.4,0.4,not fire
15,06,2012,28,80,17,3.1,49.4,3,7.4,0.4,3,0.1,not fire
```

Figura 3.2: Se muestran los primeros 15 registros de los incendios forestales en Dataset 'Algerian_forest_fires_dataset'.

3.2. Tratamiento de los datos

Una vez se han descrito los dos conjuntos de datos con los que vamos a trabajar, vamos a proceder a tratarlos. Cada conjunto de datos se va a limpiar y procesar de forma independiente para posteriormente, una vez ya sean compatibles, conjuntarlos en un solo Dataset. Los pasos que se van a seguir son los descritos en el capítulo 2, en la sección Tratamiento de los datos.

3.2.1. Eliminación de los duplicados y procesamiento de tipos

Para la eliminación de los duplicados, se ha utilizado la función 'quitarDuplicados'. Esta función recibe como parámetro el dataframe con el conjunto de datos y devuelve el mismo Dataframe pero habiendo eliminado los registros duplicados. En ambos Datasets, no se han detectado ningún registro duplicado.

Para el correcto tratamiento de los datos, se utiliza la función 'tratarString'. Esta función elimina cualquier espacio en los nombres de las columnas. De esta manera, se elimina cualquier la posibilidad de haber alguna confusión.

Antes de procesar los tipos, tenemos que analizar de qué tipo son las columnas de cada Dataset. Analicemos primero el tipo de los datos del Dataset de Portugal: Como se puede ver en la imagen, todos los datos son de tipo numérico, excepto las columnas month y day. Esto se corresponde a que los datos de estas columnas contienen las primeras tres letras del mes y el día, respectivamente. Por lo tanto, como los modelos no procesan datos categóricos, se transforman estas columnas al mes y día correspondiente en número.

```
Data columns (total 13 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   X       517 non-null    int64
 1   Y       517 non-null    int64
 2   month   517 non-null    object
 3   day     517 non-null    object
 4   FFMC    517 non-null    float64
 5   DMC     517 non-null    float64
 6   DC      517 non-null    float64
 7   ISI     517 non-null    float64
 8   temp    517 non-null    float64
 9   RH      517 non-null    int64
 10  wind    517 non-null    float64
 11  rain    517 non-null    float64
 12  area    517 non-null    float64
```

Figura 3.3: Representación de los tipos de los datos del dataset 'forestfires'.

Ahora veamos el tipo de los datos del conjunto de datos de Argelia: Se puede observar que solo una columna es de tipo categórico, mientras que el resto son de tipo numérico. Para tratar esta columna Class, se han cambiado los valores 'not fire' por 0 y los valores 'fire' por 1. De esta manera, se pueden introducir los datos en el modelo ya que todos son de tipo numérico.

3.2.2. Tratamiento de campos nulos

En esta sección, se va a explicar el tratamiento de los campos nulos. En ambos conjuntos de datos, se ha comprobado que no hay ningún campo nulo. Sin embargo, se propone la función 'reemplazarNulos', en caso de que los hubiera. Esta función cuenta el número de campos nulos que hay por columna. Si hay más de un umbral, que se le pasa como parámetro, medido en porcentaje, se eliminan directamente. Esto permite eliminar las columnas con gran porcentaje de nulos, o simplemente ninguna, según el caso de uso. Respecto al tratamiento del resto de campos nulos, se reemplazan por valores nuevos utilizando la sustitución por la distribución de los datos de cada columna original. Mediante el uso de las frecuencias originales relativas, se generan datos nuevos que reemplazan los valores nulos y así no se modifica la distribución de los datos.

```
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   day          244 non-null    int64
 1   month        244 non-null    int64
 2   year         244 non-null    int64
 3   Temperature  244 non-null    int64
 4   RH           244 non-null    int64
 5   Ws           244 non-null    int64
 6   Rain         244 non-null    float64
 7   FFMC         244 non-null    float64
 8   DMC          244 non-null    float64
 9   DC           244 non-null    float64
 10  ISI          244 non-null    float64
 11  BUI          244 non-null    float64
 12  FWI          244 non-null    float64
 13  Classes      244 non-null    object
```

Figura 3.4: Representación de los tipos de los datos del dataset $'Algerian\_forest\_fires\_dataset'$.

3.2.3. Normalización

Cómo no tenemos ningún campo de tipo categórico en ninguno de los dos conjuntos de datos, nos vamos a centrar solo en la normalización de los campos numéricos de ambos conjuntos. Para ello, se utiliza la función 'normalizarDatos'. Esta función toma las columnas numéricas del conjunto de datos que se le pasa por parámetro, en nuestro caso, todas las columnas son numéricas. Después, utiliza Min-Max Scaling para transformar los datos al rango [0,1]. En realidad, toma el valor mínimo, que será 0, y el máximo, que será 1, y escala el resto de valores intermedios al rango (0,1). El resultado es:

	Day	Month	Temp	RH	Wind	Rain	FFMC	DMC	DC	ISI	Class
0	1	6	29	57	18	0.0	65.7	3.4	7.6	1.3	0
1	2	6	29	61	13	1.3	64.4	4.1	7.6	1.0	0
2	3	6	26	82	22	13.1	47.1	2.5	7.1	0.3	0
3	4	6	25	89	13	2.5	28.6	1.3	6.9	0.0	0
4	5	6	27	77	16	0.0	64.8	3.0	14.2	1.2	0

Figura 3.5: Conjunto de datos $'Algerian\_forest\_fires\_dataset'$ normalizado.

3.2.4. Outliers

A continuación, vamos a estudiar los valores atípicos de cada conjunto de datos. Para ello, se va a usar la función 'detectarOutliers'. Esta función toma el Dataframe, que se le pasa por parámetro, y calcula los outliers que hay en cada columna.

Para ello, utiliza el rango intercuartílico (IQR), que mide la dispersión entre el primer y tercer cuartíl tratándolos como límites inferior y superior, respectivamente.

Ahora, veamos los outliers de cada conjunto de datos y cómo tratarlos. En el Dataset de Portugal, se han encontrado los siguientes outliers por columna:

- Month: 76 outliers.
- Day: 0 outliers.
- FFMC: 53 outliers.
- DMC: 17 outliers.
- DC: 8 outliers.
- ISI: 23 outliers.
- Temp: 2 outliers.
- RH: 12 outliers.
- Wind: 13 outliers.
- Rain: 8 outliers.
- Class: 0 outliers.

Como se puede ver, las columnas donde destacan los valores atípicos son Month, FFMC, DMC, ISI, Rh y Wind. Los outliers de la columna se deb al tratamiento realizado sobre la columna Month y Day al convertirlos a variables numéricas. El resto de outliers se deben a que la medición del clima y de los factores ambientales no siempre es predecible y no tiene porqué seguir una distribución concreta. Además, es posible que haya habido algún error de medición o al introducir los datos.

Y, en el conjunto de datos argelino, se han encnotrado por columna los siguientes valores atípicos:

- Month: 0 outliers.
- Day: 0 outliers.
- FFMC: 16 outliers.
- DMC: 12 outliers.

- DC: 15 outliers.
- ISI: 4 outliers.
- Temp: 2 outliers.
- RH: 0 outliers.
- Wind: 8 outliers.
- Rain: 35 outliers.
- Class: 0 outliers.

Como se puede observar en la lista, las columnas donde destacan los valores atípicos son FFMC, DMC, DC y Rain. De manera análoga, estos valores atípicos se deben a que la medición del clima y de los factores ambientales no siempre sigue una distribución concreta, y a posibles errores que haya habido en la medición e introducción de los datos. Por estas razones, y porque el conjunto de datos es de pequeño tamaño, se ha decidido no eliminar los outliers. De esta manera, no se pierde información valiosa para el modelo.

3.2.5. Relación entre las categorías

Para estudiar la relación de las columnas de los conjuntos, hay que distinguir entre los tipos de datos de cada columna. En nuestro caso, las columnas de ambos Dataframes son todas numéricas, por lo que solo usaremos la correlación de Pearson para el estudio. cabe destacar que se va a estudiar la correlación entre la columna Class y el resto de columnas, ya que nos interesa estudiar la relación de las columnas en la variable objetivo. Para este estudio se usa la función 'correlacionPearson'. Esta función toma el Dataframe que se le pasa como parámetro y calcula la correlación de Pearson entre la columna Class y el resto de columnas. Devuelve las correlaciones correspondientes.

En el caso de que el Dataframe también tenga varias categóricas, se ha implementado también la función 'columnasRelacionadas'. Esta función toma dos columnas de un dataframe y estudia su relación. Para ello, distingue si las columnas son numéricas o categóricas. Si ambas son numéricas, se estudia la correlación de Pearson; si ambas son categóricas, se estudia la prueba chi cuadrado; y, si una es numérica y otra categórica, se estudia el test ANOVA entre las columnas. Cada prueba devuelve el pvalor, que luego es convertido a True, si es menor a 0.05, y False, si es mayor. De esta manera, la función devuelve True si las columnas están relacionadas y False si no lo están.

En nuestro caso de uso, veamos la correlación entre las columnas del conjunto de datos de Portugal:

- Month: 0.0867
- Day: -0.0126
- FFMC: 0.0355
- DMC: 0.0583
- DC: 0.0806
- ISI: 0.0379
- Temp: 0.0811
- RH: -0.0181
- Wind: -0.0660
- Rain: 0.0032

Se puede apreciar que la correlación es baja.Tras un estudio de los datos, se concluye que esto se debe a que la gran mayoría de los registros son incendios, es decir, en la columna Class, la mayor parte de los registros son 1.

Veamos ahora la correlación en el conjunto de datos argelino:

- Month: 0.0223
- Day: 0.2018
- FFMC: 0.7701
- DMC: 0.5842
- DC: 0.5071
- ISI: 0.7361
- Temp: 0.5181
- RH: -0.4350
- Wind: -0.0665

- Rain: -0.3794

Comentemos los datos obtenidos. Podemos observar que la ocurrencia de un incendio se debe en mayor medida al FFMC, que es un índice de la ignición de los combustibles finos de la superficie. También está muy relacionado con el ISI, que se corresponde con el índice de propagación del incendio. Le siguen la temperatura, el índice de sequía y el contenido de humedad en el suelo. Cabe destacar la relación negativa con la humedad ambiente y la lluvia. Cabe destacar también que el viento parece no influir.

Tras este análisis, se observa que columnas como el viento o el mes no tienen casi correlación con la provocación de incendios. Sin embargo, dada la poca cantidad de variables o columnas de las que se dispone, se ha decidido mantenerlas.

3.2.6. Cifrado de datos

En estos dos conjuntos de datos, no se encuentran datos confidenciales que haga falta encriptar. Igualmente, si hubiera dicha necesidad, se puede utilizar el cifrado propuesto en el capítulo anterior.

Si hubiera la necesidad de cifrar los datos al ser sensibles o confidenciales, se proponen las funciones 'encriptador', para cifrar los datos, y 'desencriptador', para descifrar los datos. La función 'encriptador' toma el Dataframe, las columnas y el salt, que se le pasan como parámetro y devuelve el Dataframe con las columnas que se le han pasado cifradas. Para cifrar, toma los campos de las columnas, les añade el salt y los cifra utilizando el algoritmo hash SHA256. Además, realiza un diccionario con los valores cifrados y el valor original para su posterior descifrado si fuera necesario. Para descifrar, se usa la función 'desencriptador'. Esta función recibe el Dataframe con las columnas cifradas y el diccionario antes descrito. La función reemplaza los campos de las columnas que aparecen en el diccionario por sus valores originales, devolviendo el Dataframe descifrado.

3.2.7. Generación de los Datos Sintéticos

En este proyecto, dado el tamaño pequeño de los datos de los que se disponen, se van a generar datos sintéticos con el objetivo de generar un mayor volumen de datos para el entrenamiento. Para ello, se va a utilizar el método GAN, el cual usa redes neuronales para generar los datos. Utilizamos la función 'generarDatosSintéticos'. Esta función utiliza los datos originales para entrenar y ajustar el modelo GAN. Finalmente, genera y devuelve el número de registros que se pidan. Los datos que devuelve siguen la misma estructura y tienen las mismas características que los

iniciales. El proceso que se ha seguido ha sido generar 5000 registros a partir de los datos originales portugueses, sin tratar, y generar 5000 registros a partir de los datos originales argelinos, también sin tratar. Después, se han ajustado los datos originales y sintéticos en cada caso, y se han aplicado todas las transformaciones de tratamiento de datos antes descritas.

3.2.8. Unión de los Dataframes y Separación de los datos

Para poder trabajar con los dos conjuntos de datos simultáneamente, vamos a unirlos en un solo Dataframe. Para que ambos sean compatibles, se han realizado una serie de transformaciones en ambos. En el caso de conjunto portugués, se han renombrado y reorganizado las columnas para que ambos Dataframes contengan las mismas. Las columnas y su orden son: 'Day', 'Month', 'Temp', 'RH', 'Wind', 'Rain', 'FFMC', 'DMC', 'DC', 'ISI' y 'Class'. Además, se han eliminado las columnas X, Y. Finalmente, para la predicción del modelo es crucial que este conjunto de datos tenga la columna Class con valores 0 y 1 que indiquen si ha habido incendio o no. Para ello, se ha transformado la columna original area, ahora llamada Class, utilizando las hectáreas quemadas por el incendio.

En cuanto al conjunto de datos argelino, también se han renombrado y reorganizado las columnas para que ambos conjuntos tengan el mismo orden y nombre en las columnas. Las columnas finales son las antes mencionadas. También se han eliminado las columnas Year, FWI y BUI.

Una vez tenemos ambos conjuntos compatibles, es decir, con las mismas columnas y con el mismo tipo de datos, los unimos en uno solo. Con lo cual, obtenemos un solo conjunto, unión de los conjuntos originales de cada origen y de los conjuntos generados sintéticamente de cada origen. En total, 10762 registros de incendios forestales con las siguientes características: 'Day', 'Month', 'Temp', 'RH', 'Wind', 'Rain', 'FFMC', 'DMC', 'DC', 'ISI' y 'Class'.

Para terminar con el tratamiento de los datos, se ha separado el Dataframe en dos conjuntos, de entrenamiento y el de prueba, representando el 80 % y 20 %, respectivamente, del conjunto total de los datos. Esta técnica escogida, requiere del uso de validación cruzada más adelante.

3.3. Modelos

Una vez tenemos listo el conjunto de datos para el entrenamiento de los modelos, es hora de implementar los modelos. Los modelos que se han implementado son

los cuatro que se han descrito en el capítulo anterior: Regresión Logística, Random Forest, Perceptrón Multicapa y K Nearest Neighbours. Cada modelo se ha implementado en las funciones 'regLog', 'randforest', 'percMulti' y 'kNearestNeighbours' respectivamente. Cada función crea el modelo, lo entrena con el conjunto de datos de entrenamiento y prueba el modelo con el conjunto de datos de prueba. Este conjunto de datos de prueba se usa para calcular la precisión del modelo. Los modelos, una vez entrenados, sirven para introducir los registros, en nuestro caso, los fenómenos ambientales del día que queramos, y los modelos devuelven su previsión del porcentaje de que ocurra un incendio.

3.4. Afinamiento de los modelos

Ahora, es el turno de la afinación de los modelos realizados, para mejorar así su capacidad de predicción. Se van a seguir los pasos descritos en el capítulo anterior.

3.4.1. Excatitud y Precisión

Vamos a evaluar los resultados obtenido por el modelo. Para ello, tras hacer el primer entrenamiento de los modelos con el conjunto de datos tratados, obtenemos las siguientes métricas de exactitud de cada modelo:

Cuadro 3.1: Comparación de la exactitud de los modelos

Modelo	**Exactitud**
Regresión Logística	98 %
Random Forest	98 %
Perceptrón Multicapa	97 %
K Nearest Neighbors	97 %

Como se puede observar, se han obtenido muy buenos resultado, ningún modelo baja del 97 %. Se aprecia que los mejores modelos son, por poco, la regresión logística y el random forest. Esto quiere decir que los modelos predicen al $97\,\% - 98\,\%$ correctamente la ocurrencia de incendios.

Obtengamos ahora la precisión de los modelos. Recordemos que se calcula tomando las predicciones positivas correctas y dividiéndolas entre la suma entre las predicciones positivas correctas y las predicciones positivas incorrectas.

Cuadro 3.2: Comparación de la precisión de los modelos

Modelo	**Precisión**
Regresión Logística	99 %
Random Forest	99 %
Perceptrón Multicapa	98 %
K Nearest Neighbors	98 %

Como podemos observar, se obtienen precisiones de entorno al $98\,\% - 99\,\%$, siendo los mejores modelos de nuevo la regresión logística y el random forest, ambos con 99 % de precisión. La precisión mide la capacidad del modelo para no cometer falsos positivos. Por lo que, los modelos realizados cometen muy pocos errores en sus predicciones.

3.4.2. Matriz de confusión

La matriz de confusión expone de forma gráfica cómo el modelo clasifica correctamente o incorrectamente las distintas clases predichas por el modelo. De esta manera, podemos ver de forma gráfica si el modelo ha clasificado correctamente o no cada predicción. En las matrices de confusión que obtenemos a través de los modelos, las filas representan la característica real de los datos y las columnas la característica predicha por el modelo.

De esta manera, la primera celda de la primera fila, representa los incendios producidos que el modelo a predicho que iban a ocurrir, la segunda celda de la primer fila representa los incendios que ha predicho el modelo que no iban a ocurrir y en la realidad sí han ocurrido, la primer celda de la segunda fila indica los incendios que en la realidad no han ocurrido y el modelo ha predicho que sí iban a ocurrir, y la segunda celda de la segunda fila indica los incendios que no han ocurrido ue así ha predicho el modelo. Veamos y comentemos las matrices de confusión de cada modelo.

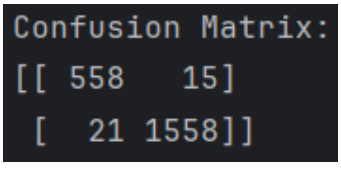

Figura 3.6: Matriz de confusión del modelo de regresión logística.

Como se puede apreciar en la matriz de confusión del modelo, el modelo ha predicho correctamente la mayoría de los registros. Solo hay 21 registros que se han predicho como incendios y no han ocurrido, y 15 que se ha predicho que no iban a tener lugar y sí han sucedido.

En cuanto al modelo de random forest, vemos que la matriz de confusión del

```
Confusion Matrix:
[[ 560   13]
 [  11 1568]]
```

Figura 3.7: Matriz de confusión del modelo de random forest.

modelo ha predicho correctamente la mayoría de los registros. Tan solo hay 11 registros que se han predicho cómo incendios y no han ocurrido, y 13 que se han predicho que no iban a tener lugar y sí han sucedido.

En cuanto a la red neuronal perceptrón multicapa, vemos que la matriz de con-

```
Confusion Matrix:
[[ 513   23]
 [  14 1603]]
```

Figura 3.8: Matriz de confusión del modelo perceptrón multicapa.

fusión del modelo ha predicho correctamente la mayoría de los registros. Tan solo hay 14 registros que se han predicho cómo incendios y no han ocurrido, y 23 que se han predicho que no iban a tener lugar y sí han sucedido.

Finalemente, para el modelo k nearest neighbors, vemos a través de la matriz

```
Confusion Matrix:
[[ 511   25]
 [  25 1592]]
```

Figura 3.9: Matriz de confusión del modelo k nearest neighbors.

de confusión que la mayoría de los registros han sido predichos correctamente. Tan

solo hay 25 registros que se han predicho cómo incendios y no han ocurrido, y 25 que se han predicho que no iban a tener lugar y sí han sucedido.

Por lo tanto, vemos que los modelos realizan, en su gran mayoría, predicciones correctas a partir del conjunto de datos.

3.4.3. Curva ROC

Recordemos que la curva ROC ilustra gráficamente la tasa de verdaderos positivos frente a la de falsos positivos. La interpretación gráfica consiste en que cuanto mejor es el modelo menos falsos positivos hay, lo que se traduce gráficamente en que la función es muy próxima a $y = 1$ cuando $x = 0$. Veamos las curvas de los cuatro modelos y comparemos:

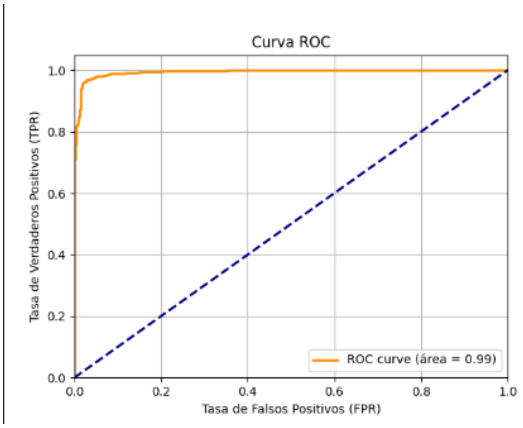

Figura 3.10: Representación de la curva ROC del modelo regresión logística.

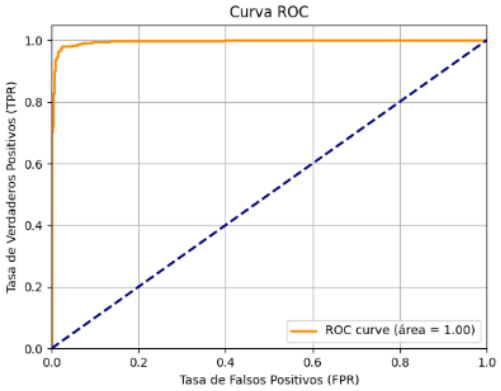

Figura 3.11: Representación de la curva ROC del modelo random forest.

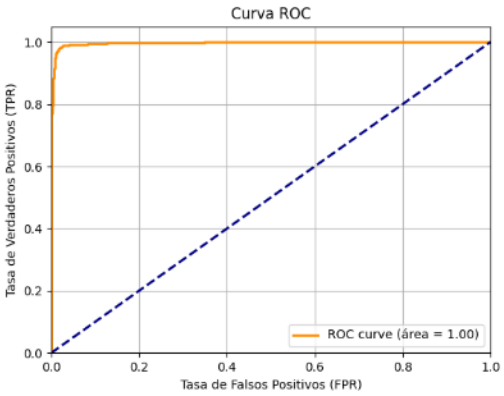

Figura 3.12: Representación de la curva ROC del modelo perceptrón multicapa.

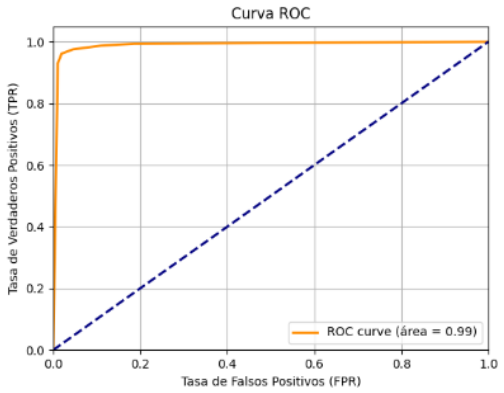

Figura 3.13: Representación de la curva ROC del modelo k nearest neighbors.

Se puede apreciar que en los cuatro modelos, la tasa de falsos positivos es muy baja. Los mejores modelos son random forest y perceptrón multicapa. Los modelos prevén, en general, correctamente las categorías.

3.4.4. Validación Cruzada

Anteriormente, se ha utilizado la técnica del Train-Test Split para la separación de los datos y el entrenamiento y evaluación de los modelos. Sin embargo, dado que la técnica K-Folds proporciona mejores resultados, ya que garantiza que todos los datos se usen alguna vez tanto en el conjunto de entrenamiento como en el conjunto de prueba, se va a utilizar esta técnica también. Mediante esta técnica, obtenemos resultados más fiables del modelo. Para ello, se utilizan 5 folds o pliegues. Obtenemos los siguientes resultados de exactitud tras utilizar validación cruzada K-folds sobre cada modelo:

Cuadro 3.3: Comparación de la exactitud de los modelos

Modelo	**Exactitud**
Regresión Logística	97 %
Random Forest	98 %
Perceptrón Multicapa	98 %
K Nearest Neighbors	97 %

Se puede observar que la exactitud de los modelos no varía mucho. Sin embargo, cabe destacar que el modelo random forests se mantiene con 98 % de exactitud, lo que quiere decir que es un modelo robusto, y el modelo perceptrón multicapa sube hasta el 98 % de exactitud también, esto se debe a que cross validation reduce la varianza y por ello, la exactitud sube en este modelo.

3.4.5. Ajuste de los hiperparámetros

Para cada modelo, se van a definir los posibles hiperparámetros junto con una lista de valores que pueden tomar. Después se va a evaluar cada combinación de cada uno con grid search o búsqueda de cuadrícula, es una técnica exhaustiva que da buenos resultados. Una vez evaluada cada opción, se toman los mejores hiperparámetros para cada modelo y se introducen en cada modelo dichos parámetros para el reentrenamiento de los modelos. Los mejores hiperparámetros son con los que se obtiene la máxima exactitud.

Los hiperparámetros que se evalúan en la regresión logística junto con los posibles valores son:

- C: con los siguienets valores: 0.01, 0.1, 1, 10.
- penalty: que puede tomar los valores l1, l2, elasticnet, None.
- solver: con las siguientes opciones: saga, lbfgs, liblinear.

Y, los hiperparámetros que dan mejores resultados tras la búsqueda por cuadrícula son: C = 1, penalty = l1 y solver = liblinear.

Veamos ahora los hiperpámetros y sus distintas opciones de cada modelo. Para el modelo random forest:

- n_estimators: con posibles valores 50, 100, 200.
- max_depth: con opciones None, 10, 20, 30.

- min_samples_split: con los siguientes valores 2, 5, 10.
- min_samples_leaf: con posibles opciones 1, 2, 4.
- bootstrap: con posibilidad de ser True o False.

Con hiperparámetros que dan mejores resultados: n_estimators = 200, max_depth = None, min_samples_leaf = 1, min_samples_split = 5 y bootstrap = False.

Ahora, para la red neuronal perceptrón multicapa, los hiperparámetros que se van a estudiar y sus posibles valores son los siguientes:

- hidden_layer_sizes: puede tomar los valores(50,), (100,), (50,50).
- activation: puede ser del tipo relu, tanh o logistic.
- solver: toma o el valor adam o sgd.
- alpha: puede tomar las opciones 0.0001 o 0.001.
- learning_rate: tiene dos posibles valores, o constant, o adaptive.

La combinación de hiperparámetros que genera los mejores resultados para el modelo son: hidden_layer_sizes = (100,), activation = relu, solver = adam, alpha = 0.0001 y learning_rate = constant.

Veamos cuáles son los hiperparámetros para el modelo K nearest neighbours, y las distintas opciones que pueden tomar:

- n_neighbors: puede tomar los valores 3, 5, 7 o 9.
- weights: pudiendo ser o uniform o distance.
- metric: los posibles valores son euclidean, manhattan y minkowski.

Por último, los hiperparámetros que mejoran la exactitud al máximo son los siguientes: n_neighbors = 9, weights = distance y metric = manhattan.

Una vez se han obtenido los mejores hiperparámetros para cada modelo, que son con los que se obtiene la máxima exactitud, se introducen en cada modelo para que, en los próximos entrenamientos, cada modelo utilice estos parámetros en sus predicciones.

3.4.6. Reentreno de los modelos

En esta sección, se va a volver a entrenar los modelos. La diferencia es que ahora se van a utilizar los hiperparámetros calculados en la sección anterior para cada modelo. Después, se van a volver a evaluar los resultados acorde al capítulo 2, aunque esta vez, de manera más breve.

Tras realizar el reentreno, hemos obtenido los siguientes resultados. En cuanto a la exactitud y la precisión, los resultados son:

Cuadro 3.4: Comparación de la exactitud de los modelos

Modelo	**Exactitud**
Regresión Logística	99 %
Random Forest	99 %
Perceptrón Multicapa	99 %
K Nearest Neighbors	99 %

Cuadro 3.5: Comparación de la precisión de los modelos

Modelo	**Precisión**
Regresión Logística	99 %
Random Forest	99 %
Perceptrón Multicapa	99 %
K Nearest Neighbors	99 %

Como se puede observar, han mejorado hasta llegar todos al 99 %. No se lleva al 100 % ya que es muy difícil obtener un modelo perfecto.

Ahora, veamos la matriz de confusión y la curva ROC de cada modelo:

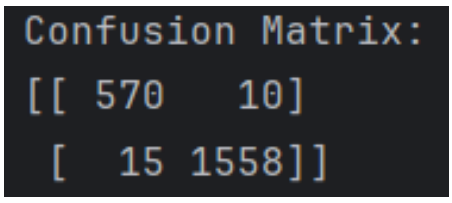

Figura 3.14: Representación de la matriz de confusión del modelo regresión logística.

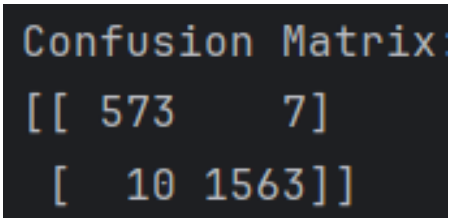

Figura 3.15: Representación de la matriz de confusión del modelo random forest.

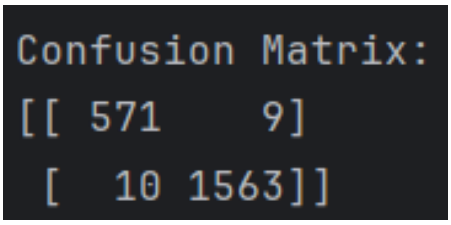

Figura 3.16: Representación de la matriz de confusión del modelo perceptrón multicapa.

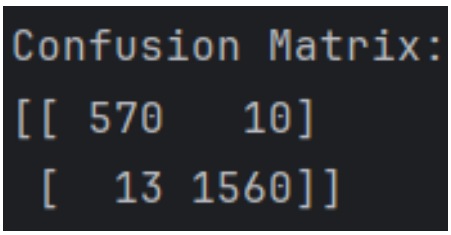

Figura 3.17: Representación de la matriz de confusión del modelo K nearest neighbors.

Vemos que los falsos positivos y los falsos negativos han descendido acorde a la precisión obtenida. Y, las curvas ROC de cada modelo son:

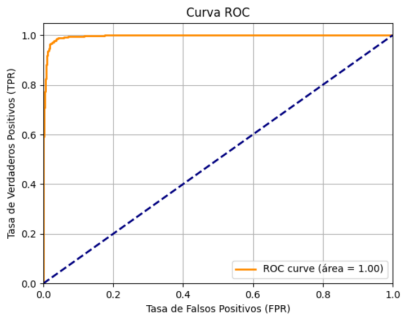

Figura 3.18: Representación de la curva ROC del modelo regresión logística.

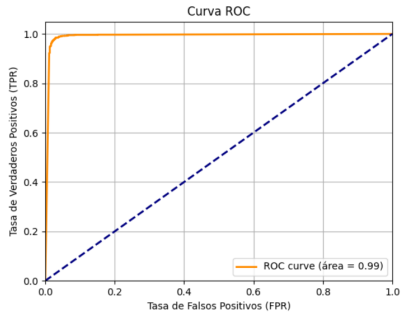

Figura 3.19: Representación de la curva ROC del modelo random forest.

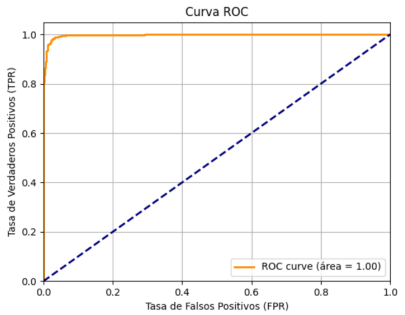

Figura 3.20: Representación de la curva ROC del modelo perceptrón multicapa.

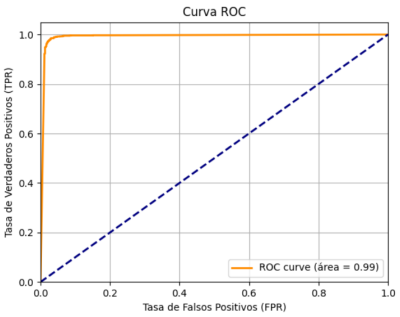

Figura 3.21: Representación de la curva ROC del modelo K nearest neighbors.

Al igual que en las matrices de confusión, vemos que ha mejorado con respecto al primer entrenamiento.

Finalmente, tras aplicar validación cruzada, se obtiene la exactitud final de los modelos:

Cuadro 3.6: Comparación de la exactitud de los modelos

Modelo	**Exactitud**
Regresión Logística	99 %
Random Forest	99 %
Perceptrón Multicapa	98 %
K Nearest Neighbors	98 %

Como se puede observar a lo largo del estudio de los datos, en general, los modelos de regresión logística y random forest, por poco, son los mejores igual que ahora en exactitud tras validación cruzada. También, se aprecia que la exactitud ha bajado en el resto de modelos, lo cuál es normal al aplicar validación cruzada, ya que se analizan todos los registros varias veces reduciendo el sesgo del analisis.

Capítulo 4

Conclusiones

Tras los resultados presentados en la sección anterior, vamos a analizarlos. Los cuatro modelos presentan muy buenos resultados de precisión y exactitud por encima del 97 %. A continuación, se van a analizar las causas de estos resultados.

El tratamiento de los datos ha sido una parte importante del estudio que ha influido en la obtención de los resultados. Tras las transformaciones que se han hecho, han permitido que los modelos traten de forma más sencilla los datos. También , ha contribuido el afinamiento de los modelos. Al ajustar los modelos con sus mejores hiperparámetros, hemos observado una mejora de la precisión y la exactitud. También, la generación de datos ha sido un punto clave, quizás sea lo que más ha contribuido a la mejora de los modelos. Ha permitido tener un conjunto de datos mayor, lo que se traduce en un modelo mejor entrenado y con menos sesgo. Además, los datos generados se han basado en la distribución de los datos originales, por lo que se han producido datos, que siguen a la perfección dicha distribución. Esto provoca que los datos sintéticos representen a los datos originales, y que a los modelos les sea más fácil analizar los datos, ya que siguen esas distribuciones. Se ha realizado un reentreno utilizando solo los datos originales y se ha observado que todos los modelos bajan en exactitud por debajo del 90 %, excepto el modelo random forest, que se mantiene con un 98 % de exactitud.

Respecto a los modelos, en la mayoría de los análisis realizados, los modelos de regresión logística y random forest son con los que se obtienen mejores resultados. Sin embargo, aunque el modelo de regresión logística es muy competitivo, el modelo random forest ha sido siempre más constante a lo largo de todas las pruebas. Es el que menor tasa de falsos positivos y negativos tiene, da buenos resultados tras la validación cruzada y mejora con el ajuste de los hiperparámetros. La única desventaja de este modelo es su coste computacional, ya que es alto debido a la creación de todos los árboles de decisión. Aunque precisamente, por ser tan

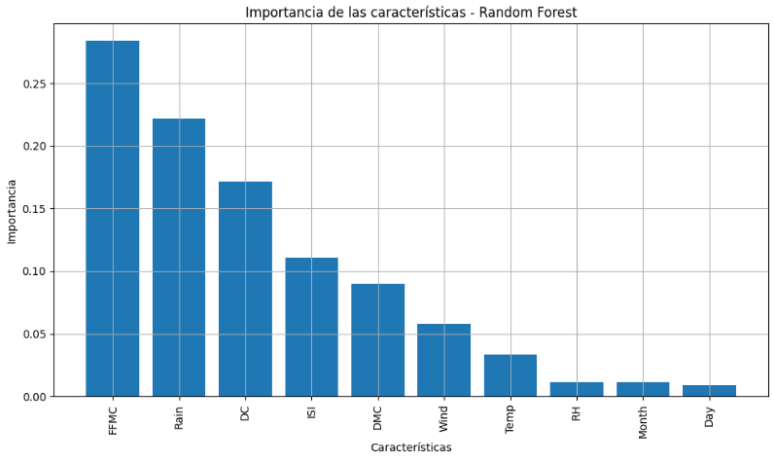

Figura 4.1: Representación de las características que más han influido en las predicciones del modelo.

exhaustivo, presenta tan buenos resultados. Además, como hemos visto, con los datos originales es el único modelo que mantiene su exactitud, lo que lo convierte en el modelo más robusto de los modelos presentados.

Por último, veamos cuáles son las variables del conjunto de datos que más influyen en la ocurrencia de los incendios. Este análisis se va a realizar solo en el modelo random forest. Las características que más influyen son FFMC, Rain y DC, con más de un 15 %, siendo la mayor FFMC con 28 % de influencia sobre las predicciones. Esto confirma que lo que más influye en la ocurrencia de un incendio o no es el índice de probabilidad de ignición, la lluvia y el índice de la sequía. Lo que quiere decir que cuanto más secos estén los combustibles finos y de gran tamaño del suelo y cuanta menos lluvia haya, más probabilidad de haber un incendio forestal hay. La siguiente categoría que influye pero en menor medida es el ISI, que se corresponde con el índice de propagación inicial. Finalmente, como cabía esperar, las características que menos influyen son los días y los meses. Es sorprendente que tanto la humedad en el aire como la temperatura influyan tan poco, menos de un 5 %. Este análisis concluye que lo que influye en gran medida en la ocurrencia de incendios forestales no son en tanta medida las condiciones climáticas del momento sino la combustibilidad del suelo.

4.1. Trabajo Futuro

En esta sección se comentan algunas propuestas para futuras investigaciones o proyectos. Estas buscan ampliar y mejorar tanto los resultados como las aplicaciones de este proyecto:

- **Utilización de mayor volumen de datos reales:** como se ha visto a lo largo del estudio, el aumento de datos a través de la generación sintética ha ayudado a mejorar los resultados. Siguiendo con este enfoque, la obtención de mayor cantidad de datos reales mejoraría el modelo. También, se podrían añadir datos climáticas más variados y de diferentes regiones.

- **Utilización de datos reales periódicamente o en tiempo real:** siguiendo con el enfoque anterior, se propone programar una aplicación que tome los datos climáticas en tiempo real de las zonas que se desean estudiar y utilizar dichos datos para el estudio. De esta manera, no solo se obtendrián nuevos datos reales para el entrenamiento de los modelos, si no que se podría utilizar la aplicación de forma autonámtica para la predicción de la ocurrencia de incendios.

- **Exploración de otros modelos y técnicas:** se propone utilizar otras modelos que no se han utilizado en este proyecto. Además, tanto en el tratamiento de los datos, como en el afinamiento de los modelos y en la generación de datos, se podrián utilizar otras técnicas que no se han usado en este trabajo.

- **Predicción de las hectáreas que afectará el incendio:** añadiendo una característica en los conjuntos de datos que indique las hectáreas afectadas si se ha proudico un incendio, se podría predecir esta característica. Se propone hacer nuevos modelos que predigan el número de hectareas afectadas. Se propone que primero se use los modelos que se proponen en este estudio para predecir la ocurrencia del incendio y posteriormente, si se ha predicho dicha ocurrencia, utilizar estos nuevos modelos para predecir su extensión.

- **Predicción del equipo necesario para la extinción:** de la misma manera que en la propuesta anterior, si se añaden los datos del material y trabajadores necesarios para la extinción de cada incendio, se pueden construir modelos que predigan el equipo necesario para la extinción del incendio. De esta manera, se podría primero predecir la ocurrencia del incendio, después el número de hectáreas que va a afectar, y por último, el equipo necesario para su extinción.

- **Establecer un orden de prioridades:** se propone utilizar el porcentaje que devuelven los modelos actuales para establecer orden de prioridades en el caso de que se den varios incendios simultáneamente. De igual manera, se podrían utilizar a futuro la predicción de las hectáreas afectadas y el meterial necesario para su extinción para establacer este orden de prioridades.

Apéndice

Apéndice A: Contenido Adicional

A continuación, se presentan los contenidos adicionales asociados a la memoria. Se puede acceder a este contenido a través del repositorio habilitado por la Escuela.

El contenido adicional contiene los conjuntos de datos originales utilizados para el estudio del proyecto, llamados 'forestfires.csv' y 'Algerian_forest_fires_dataset.csv'. Contienen los registros de incendios de Portugal y Argelia, respectivamente, junto con sus características.

Además, se incluyen los scripts de Python con los códigos que se han realizado para la implementación del estudio. Estos son:

- DatosSinteticos.py: Este script contiene la función que implementa la generación de datos sintéticos utilizando el modelo GAN.
- KNearestNeighbors.py: En este script, se encuentra la función que implementa el modelo k nearest neighbors junto con sus métricas de evaluación.
- LogisticRegression.py: Este script contiene la función que implementa el modelo regresión logística junto con sus métricas de evaluación.
- PerceptronMulticapa.py: Este script contiene la función que implementa el modelo red neuronal perceptrón multicapa junto con sus métricas de evaluación.
- RandomForest.py: En este script, se encuentra la función que implementa el modelo random forest junto con sus métricas de evaluación.
- TFG.py: En este script, es donde se realiza el análisis. Primero que se leen los ficheros con la información de los incendios, se generan los datos sintéticos y se tratan los datos. Depués, se entrenan los modelos y se generan las métricas de evaluación y ajuste de sus hiperparámteros.

- TratamientoArg.py: Contiene el tratamiento de datos del conjunto de datos 'Algerian_forest_fires_dataset.csv'.
- TratamientoPor.py: Contiene el tratamiento de datos del conjunto de datos 'forestfires.csv'.
- Utils.py: Este script contiene todas las funciones que se usan para el tratamiento de los datos.

Bibliografía

[1] *Incendios forestales en España: índice de riesgo de incendios y quema de restos vegetales*, AEMET Blog, https://aemetblog.es/2022/10/30/incendiosforestales-en-espana-indice-de-riesgo-de-incendios-y-quema-de-restos-vegetales/?utm_source=.com

[2] *Modelos predictivos y prevención de incendios forestales en Castilla y León*, Bulería, Universidad de León, https://buleria.unileon.es/handle/10612/23216?utm_source=.com

[3] *La UMU desarrolla una herramienta para prever incendios forestales con un 70 % de eficacia en zonas de riesgo*, Cadena SER, https://cadenaser.com/murcia/2025/04/07/la-umu-desarrolla-una-herramienta-para-prever-incendios-forestales-con-un-70-de-eficacia-en-zonas-de-riesgo-radio-murcia/?utm_source=.com

[4] *Extreme Weather*, NASA Climate, https://science.nasa.gov/climate-change/extreme-weather/?utm_source=.com

[5] *Incendios forestales en Castilla y León: agilizar procesos y medios*, El Español, https://www.elespanol.com/castilla-y-leon/region/20230413/incendios-forestales-castilla-leon-agilizar-procesos-medios/755924662_0.html?utm_source=.com

[6] *Gestión ágil vs gestión tradicional de proyectos: cómo elegir*, Escuela de Negocios FEDA, https://www.escueladenegociosfeda.com/blog/50-la-huella-de-nuestros-docentes/471-gestion-agil-vs-gestion-tradicional-de-proyectos-como-elegir

[7] *¿Qué es la metodología clásica de desarrollo de software?*, Ginzo Tech, https://ginzo.tech/metodologia-clasica-desarrollo-software/

[8] *¿Qué es PyCharm y cómo usarlo?*, DataScientest, https://datascientest.com/es/pycharm

[9] *¿Qué es LaTeX?*, Universidad de Alicante, https://desarrolloweb.dlsi.ua.es/cursos/2015/herramientas-investigacion/que-es-latex

[10] *Overleaf: editor colaborativo en línea para documentos LaTeX*, Universidad Carlos III de Madrid, https://www.uc3m.es/sdic/articulos/2021/overleaf

[11] *Qué es Outlook 365 y cómo sacarle partido*, Cursos Femxa, https://www.cursosfemxa.es/blog/outlook-365

[12] *¿Qué hace un científico de datos y qué salidas laborales tiene?*, Revista UNIR, https://www.unir.net/revista/ingenieria/cientifico-de-datos/

[13] *Sueldo de un especialista en Machine Learning en España*, Glassdoor,https://www.glassdoor.es/Sueldos/machine-learning-sueldo-SRCH_KO0,16.htm

[14] *Salario de un Data Scientist, Data Analyst, Ingeniero de Datos y más*, Nuclio Digital School, https://nuclio.school/blog/salario-data-scientist-data-analyst-ingeniero-de-datos-analista-de-big-data/

[15] *El método PERT*, Enredando Proyectos, https://enredandoproyectos.com/el-metodo-pert/

[16] *Inteligencia Artificial: definición y aplicaciones*, DataScientest, https://datascientest.com/es/inteligencia-artificial-definicion

[17] *Ramas de la Inteligencia Artificial*, Universidad UNIE, https://www.universidadunie.com/blog/ramas-inteligencia-artificial

[18] *Artificial Intelligence*, IBM Think, https://www.ibm.com/es-es/think/topics/artificial-intelligence

[19] *Diferencias entre aprendizaje supervisado y no supervisado*, Universidad Europea, https://universidadeuropea.com/blog/aprendizaje-supervisado-no-supervisado/

[20] *Lista de pasos para un proyecto de Machine Learning*, Jose R. Zapata, https://joserzapata.github.io/post/lista-proyecto-machine-learning/#2-obtener-los-datos

[21] *¿Cómo lidiar con los valores faltantes?*, 4Geeks, https://4geeks.com/es/lesson/como-lidiar-con-los-valores-faltantes

[22] *¿Qué son los datos numéricos?*, QuestionPro, https://www.questionpro.com/blog/es/datos-numericos/

[23] *¿Qué son los datos categóricos?*, QuestionPro, https://www.questionpro.com/blog/es/datos-categoricos/

[24] Marta Casas Delgado, *Cómo identificar y tratar outliers con Python*, Medium, https://medium.com/@martacasdelg/c%C3%B3mo-identificar-y-tratar-outliers-con-python-bf7dd530fc3

[25] Microsoft Learn, *Normalize data - Azure Machine Learning*, https://learn.microsoft.com/es-es/azure/machine-learning/component-reference/normalize-data?view=azureml-api-2

[26] Scikit-learn, *Preprocessing data*, https://scikit-learn.org/stable/modules/preprocessing.html

[27] Microsoft Learn, *Cómo configurar divisiones de datos para validación cruzada*, https://learn.microsoft.com/es-es/azure/machine-learning/how-to-configure-cross-validation-data-splits?view=azureml-api-1

[28] Microsoft Learn, *Cross-validate model - Azure Machine Learning*, https://learn.microsoft.com/es-es/azure/machine-learning/component-reference/cross-validate-model?view=azureml-api-2

[29] Google Developers, *División de conjuntos de datos*, Curso de Machine Learning, https://developers.google.com/machine-learning/crash-course/overfitting/dividing-datasets?hl=es-419

[30] *Eligiendo buenas variables a través de coeficientes de correlación – Machine Learning*, Comunidad aiutechallenge, UTEC, disponible en: https://aichallenge.utec.edu.uy/community/machine-learning/eligiendo-buenas-variables-a-traves-de-coeficientes-de-correlacion/

[31] *How to get correlation between two categorical variable and a categorical variable and continuous variable?*, StackOvercoder (versión en español de StackOverflow), disponible en: https://stackovercoder.es/datascience/893/how-to-get-correlation-between-two-categorical-variable-and-a-categorical-variab

[32] Miguel Ángel Hernández Flores, *El coeficiente de correlación de Pearson con ejemplo en Python*, Medium, https://medium.com/@hdezfloresmiguelangel/el-coeficiente-de-correlaci%C3%B3n-de-pearson-con-ejemplo-en-python-6e8588f67e35

[33] Mind the Graph, *Test Chi Cuadrado: Qué es y cómo se utiliza*, https://mindthegraph.com/blog/es/chi-square-test/

[34] DATAtab, *Chi Square Test - Tutorial*, https://datatab.es/tutorial/chi-square-test

[35] Economipedia, *Estadístico F*, https://economipedia.com/definiciones/estadistico-f.html

[36] AWS, *¿Qué son los datos sintéticos?*, https://aws.amazon.com/es/what-is/synthetic-data/

[37] DataCamp, *Synthetic Data Generation: Tutorial*, https://www.datacamp.com/es/tutorial/synthetic-data-generation

[38] *¿Qué es una red generativa antagónica (GAN)?*, Amazon Web Services (AWS), https://aws.amazon.com/what-is/gan/

[39] *The Math Behind Logistic Regression*, Medium, 19 Feb. 2020, https://medium.com/analytics-vidhya/the-math-behind-logistic-regression-c2f04ca27bca

[40] *Logistic Regression*, IBM, https://www.ibm.com/es-es/think/topics/logistic-regression

[41] *Random Forest*, IBM, https://www.ibm.com/es-es/think/topics/random-forest

[42] *Random Forest: Bosque Aleatorio, Definición y Funcionamiento*, DataScientest, https://datascientest.com/es/random-forest-bosque-aleatorio-definicion-y-funcionamiento

[43] DataScientest, *¿Qué es el algoritmo KNN?*, https://datascientest.com/es/que-es-el-algoritmo-knn

[44] IBM, *¿Qué es KNN (k-vecinos más cercanos)?*, https://www.ibm.com/es-es/think/topics/knn

[45] IBM, *Redes neuronales*, https://www.ibm.com/es-es/think/topics/neural-networks

[46] DataCamp, *Multilayer Perceptrons in Machine Learning*, https://www.datacamp.com/es/tutorial/multilayer-perceptrons-in-machine-learning

[47] DataScientest, *Cross-validation: definición e importancia*, https://datascientest.com/es/cross-validation-definicion-e-importancia

[48] AWS, *¿Qué es la optimización de hiperparámetros?*, https://aws.amazon.com/es/what-is/hyperparameter-tuning/

[49] IBM, *Optimización de hiperparámetros*, https://www.ibm.com/es-es/think/topics/hyperparameter-tuning

[50] Microsoft Learn, *Cómo comprender el aprendizaje automático automatizado (AutoML)*, https://learn.microsoft.com/es-es/azure/machine-learning/how-to-understand-automated-ml?view=azureml-api-2

[51] Gonzalo Gasca, *Precisión y recuperación (Precision-Recall)*, Medium, https://medium.com/@gogasca_/precisi%C3%B3n-y-recuperaci%C3%B3n-precision-recall-dc3c92178d5b

[52] *Forest Fires Dataset*, UCI Machine Learning Repository, https://archive.ics.uci.edu/datasets?kip=0&take=10&sort=desc&orderBy=NumHits&search=Forest+Fires

[53] *Fire Weather Index (FWI)*, Climate-ADAPT, Agencia Europea del Medio Ambiente, https://climate-adapt.eea.europa.eu/es/metadata/indicators/fire-weather-index

[54] *Índice Meteorológico de Incendios Forestales (FWI)*, Ministerio para la Transición Ecológica y el Reto Demográfico (MITECO), https://www.miteco.gob.es/content/dam/miteco/es/ministerio/servicios/estadisticas/files-1/Estadisticas/23097%20IMS%20FWI_V_Intermedia_13_07_2021.pdf

[55] Unai Samper Sánchez, *FWI (Fire Weather Index): Método de valoración del peligro de incendio forestal*, LinkedIn, https://www.linkedin.com/pulse/fwi-fire-weather-index-m%C3%A9todo-de-valoraci%C3%B3n-del-un-samper-s%C3%A1nchez/

[56] Generalitat Valenciana, *Interpretación de Boletines FWI*, Servicio de Prevención de Incendios Forestales, https://prevencionincendiosgva.es/Documents/Manuales/Interpretacion_Boletines_FWI.pdf