UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería del Software

Trabajo Fin de Grado

Análisis de datos de telemetría de Fórmula 1 con técnicas de Deep Learning

David Morán Montero

Julio, 2022

UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería del Software

Trabajo Fin de Grado

Análisis de datos de telemetría de Fórmula 1 con técnicas de Deep Learning

Autor: David Morán Montero

Tutor: Alberto Gómez Mancha

Resumen

La inteligencia artificial dentro del ámbito de la Fórmula 1 está creciendo considerablemente durante estos últimos años. Desde mejorar la experiencia de los espectadores durante las carreras hasta aumentar el rendimiento del vehículo, la ingeniería de datos dentro de la Fórmula 1 es un aspecto indispensable de este deporte.

Aunque la cantidad de datos producida en cada carrera es enorme, existen pocas aplicaciones que acerquen esta información a cualquier aficionado independientemente de su nivel de informática. Es por ello que durante la realización de este trabajo, se ha intentado analizar y visualizar datos de Fórmula 1 de forma sencilla y dinámica para cualquier aficionado sin conocimientos previos sobre ingeniería de datos.

Este documento detalla el proceso de investigación, elaboración y evaluación de algoritmos y modelos de aprendizaje automático para el agrupamiento de los circuitos de Fórmula 1 y la predicción de los resultados de los pilotos durante una carrera de Fórmula 1.

Por otro lado, se crea una aplicación web que permite visualizar los datos de telemetría y las estadísticas generales de la Fórmula 1 de forma dinámica e interactiva, con el fin de que cualquier aficionado sin conocimientos previos sobre ingeniería de datos pueda entender y observar este tipo de datos.

Abstract

Artificial intelligence within the field of Formula 1 is growing considerably in recent years. From improving the spectator experience during races to increasing vehicle performance, data engineering within Formula 1 is an indispensable aspect of the sport.

Although the amount of data produced in each race is enormous, there are few applications that bring this information closer to any fan regardless of their level of computing. That is why during the development of this work, an attempt has been made to analyze and visualize Formula 1 data in a simple and dynamic way for any fan without prior knowledge of data engineering.

This document details the process of research, elaboration and evaluation of Machine Learning algorithms and models to group Formula 1 tracks and to make predictions of drivers' results during a Formula 1 race.

On the other hand, a web application is created to show telemetry data and general statistics of Formula 1 in a dynamic and interactive way, so any fan without prior knowledge of data engineering can comprehend this type of data.

Índice general

Índice de figuras

Índice de tablas

1. Introducción

La Fórmula 1 es la categoría reina del automovilismo internacional y se encuentra en la vanguardia del mundo de la ingeniería de datos. Millones de datos se recogen de los vehículos de Fórmula 1 cada segundo haciéndolo un mundo bastante intensivo en datos.

Estos datos de telemetría permiten a los equipos mejorar el rendimiento del piloto durante una sesión o conocer los problemas del vehículo y solucionarlos de forma inmediata. Aunque este deporte presenta un gran potencial dentro del mundo de la ingeniería de datos e inteligencia artificial, lo cierto es que existen pocas investigaciones al respecto.

Por esta razón, la motivación de este proyecto es estudiar y analizar los datos de telemetría de la Fórmula 1 a través de técnicas de aprendizaje automático. Durante el desarrollo de este proyecto se aplicarán algoritmos de *clustering* para el agrupamiento de los diferentes circuitos presentes en el calendario oficial de la Fórmula 1 y se construirán modelos de aprendizaje profundo que predecirán los resultados de los pilotos durante una carrera de Fórmula 1. Además, se construirá una aplicación web que permitirá visualizar los datos de telemetría y generar estadísticas generales de la Fórmula 1.

Para ello, se marcarán una serie de alcances y objetivos, se explicará el estado de la cuestión de los elementos involucrados en el desarrollo del proyecto, se evaluarán los algoritmos y modelos aplicados a los conjuntos de datos y, finalmente, se comentarán las conclusiones obtenidas sobre los resultados de la evaluación.

2. Alcance y objetivos

El alcance de este trabajo es bastante amplio, ya que se desarrolla un proyecto de desarrollo software de ciencia de datos centrado en las siguientes fases de su ciclo de vida:

- **Extracción y procesamiento de la información**: Se definen los diferentes métodos y funciones que permiten la correcta extracción de información para su posterior análisis y visualización.
- **Análisis de los datos**, a través de diferentes modelos y técnicas de aprendizaje automático.
- **Visualización de la información**: Se realiza una aplicación web que permite la correcta visualización de los datos analizados, además de mostrar información útil para el usuario: tiempos por vuelta, telemetría de un piloto, análisis de circuitos, etc.

El objetivo principal de este proyecto es analizar y visualizar datos de telemetría de Fórmula 1, con el fin de que cualquier aficionado sin conocimientos previos sobre ingeniería de datos pueda entender y observar este tipo de datos de forma sencilla y dinámica.

A continuación, se pueden distinguir los diferentes objetivos específicos (o subobjetivos) planteados para la realización de este trabajo:

- Conocer y aplicar las técnicas de aprendizaje automático (*Machine Learning*) y aprendizaje profundo (*Deep Learning*) sobre datos de telemetría de Fórmula 1. En concreto, se desea realizar dos modelos que permitan: agrupar los distintos circuitos del calendario oficial de Fórmula 1 y predecir, a través de redes neuronales artificiales, los resultados de los pilotos en una carrera concreta.

- Crear una aplicación web que permita utilizar los modelos entrenados durante el análisis de la información. Además, se desea mostrar de forma dinámica diferentes datos de telemetría que permiten realizar un análisis visual sencillo.

- Ampliar mis conocimientos sobre el uso del lenguaje de programación de Python para desarrollar modelos de ingeniería de datos e inteligencia artificial.

- Aprender nuevas tecnologías web, como React y FastAPI [24, 25], para realizar visualizaciones de datos dinámicas en cualquier dispositivo.

3. Estado de la cuestión

En este apartado se muestra la fundamentación teórica necesaria para comprender todos los aspectos del desarrollo de este trabajo.

3.1. Bases del proyecto

A continuación se presentan los temas y trabajos relacionados que definen las bases de este proyecto.

3.1.1. Fórmula 1

La Fórmula 1 (también conocida como F1) es la máxima categoría del deporte de motor y es la principal competición de automovilismo internacional. La Fórmula 1 está dirigida por la Federación Internacional del Automóvil (FIA) y es gestionada por el grupo Liberty Media.

Figura 3.1: Logo oficial de la Fórmula 1 [1].

El campeonato de Fórmula 1 consta actualmente de 20 pilotos y 10 escuderías, entre las que se encuentran reconocidas marcas de autómoviles como Ferrari,

Mercedes o McLaren. Los pilotos y los equipos compiten mediante un sistema de puntuación durante una temporada para ganar el campeonato de pilotos y el campeonato de constructores, respectivamente.

Una temporada de Fórmula 1 consiste en una serie de carreras, conocidas como Grandes Premios, las cuales tienen lugar alrededor del mundo en autódromos y circuitos callejeros. Un Gran Premio de Fórmula 1 se celebra durante un fin de semana, el cual se divide de la siguiente forma:

- Durante el viernes se realizan sesiones de entrenamiento que permiten a las escuderías y pilotos recoger información detallada del rendimiento del vehículo en el circuito.
- El sábado se celebra la clasificación, que determina la posición de salida de un piloto en la carrera.
- La carrera se disputa el domingo, donde se premia a los 10 primeros pilotos con puntos para avanzar en el campeonato.

3.1.2. Telemetría en la Fórmula 1

La telemetría es la tecnología que permite la recopilación remota de distintos datos. Se encarga de recoger, procesar y transmitir la información hasta el sistema de monitorización.

Un vehículo de Fórmula 1 presenta más de 300 sensores que son gestionados a través de un pequeño ordenador denominado Standard Electronic Control Unit (SECU) [26]. Los sensores ayudan a monitorizar, controlar y optimizar al vehículo y al piloto mediante la recolección de datos sobre frenadas, velocidad de curva, caja de cambios, rotación del volante o temperatura de neumáticos, entre otros

muchos datos. Estos datos son gestionados por los ingenieros de equipo, que pueden resolver los diferentes problemas de forma remota y mejorar la eficiencia del vehículo.

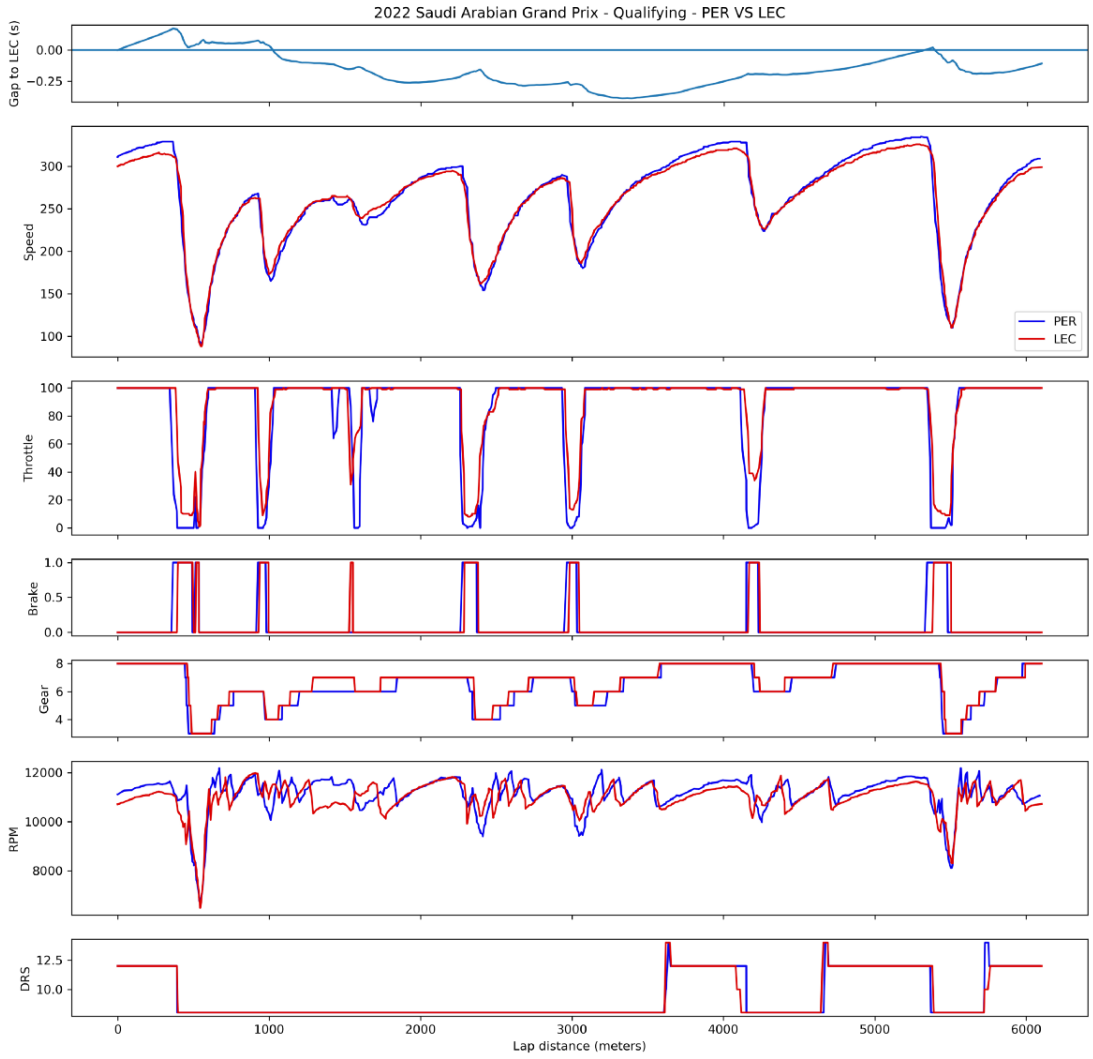

Figura 3.2: Telemetría asociada a dos pilotos durante una vuelta a un circuito [2].

Además, la visualización de estos datos, como se puede observar en la figura 3.2, permite a los pilotos y escuderías comprender el rendimiento del vehículo para mejorar el tiempo por vuelta.

3.2. Fases de un proyecto de ciencia de datos

La ciencia de datos es el campo de la informática que utiliza métodos, procesos y algoritmos para obtener conocimiento a partir de datos estructurados y no estructurados. El conocimiento obtenido se atribuye a una gran variedad de aplicaciones, desde la detección de tumores en el sector de la salud hasta la automatización del marketing digital [27].

Según [28], el ciclo de vida de un proyecto de ciencia de datos consta de las siguientes fases:

Figura 3.3: Ciclo de vida de un proyecto de ciencia de datos.

3.2.1. Definición del problema

El primer paso de un proyecto de ciencia de datos es entender el problema que se va a intentar resolver y definir los objetivos del proyecto. Un mejor entendimiento del problema incrementa las posibilidades de construir un modelo o producto basado en datos que afecte de forma positiva en una organización.

3.2.2. Extracción de datos

La recolección de datos es esencial para construir un buen modelo que permita alcanzar los objetivos propuestos. Esta información puede ser extraída de múltiples fuentes.

3.2.3. Procesamiento de la información

Una vez se hayan recopilado los datos necesarios, se deberá organizar, limpiar y procesar estos datos para mejorar la calidad del modelo a construir y evitar inconsistencias en su desarrollo.

3.2.4. Análisis exploratorio de los datos

El análisis exploratorio de los datos permite inspeccionar profundamente todas las características y propiedades de los datos, además de crear relaciones entre variables. Un análisis efectivo de los datos proporciona una base adecuada para construir modelos de gran eficacia.

3.2.5. Desarrollo y evaluación del modelo

En esta fase, se realiza el modelado de los datos mediante la construcción y evaluación de modelos predictivos. Para la evaluación del modelo se utilizan distintas métricas que permiten comprobar los resultados dependiendo del tipo de problema que se quiere solucionar.

3.2.6. Comunicación de los resultados

Una vez construido y evaluado el modelo, se deben comunicar los resultados a través de documentos técnicos. Es bastante importante la presentación de los resultados obtenidos, ya que permite entender cómo el modelo resuelve el problema definido en la primera fase del ciclo de vida.

3.2.7. Despliegue del modelo

Finalmente, se despliega el modelo o producto basado en datos de cara al público. Tras el despliegue, es necesario que exista un mantenimiento del software que evalúe el modelo con el tiempo.

3.3. Inteligencia artificial

La inteligencia artificial es la disciplina que tiene como propósito el desarrollo de sistemas inteligentes para imitar el proceso del pensamiento humano. La definición completa de inteligencia artificial sigue cuatro enfoques diferentes [29]: dos centrados en los humanos (sistemas que piensan como humanos, y sistemas que actúan como humanos) y dos centrados en torno a la racionalidad (sistemas que piensan racionalmente y sistemas que actúan racionalmente).

El término de inteligencia artificial nació en el año 1956 durante una conferencia en la Universidad de Darthmouth, donde el informático John McCarthy pronunció las siguientes palabras: *"la ciencia y la ingeniería de crear máquinas inteligentes, especialmente programas de computación inteligentes"* [30]. Durante los años 60, 70 y 80, la inteligencia artificial fue evolucionando lentamente en

laboratorios de investigación académica, debido a la falta de financiación y desarrollo de nuevas tecnologías [31] que permitieran el correcto desarrollo de la misma.

A partir de los años 90, la inteligencia artificial entró en auge gracias al repentino desarrollo de las tecnologías de la industria. Un hito importante de este campo sucedió en 1997 a manos de *Deep Blue* [32], la inteligencia artificial que ganó al campeón del mundo de ajedrez, Garry Kasparov.

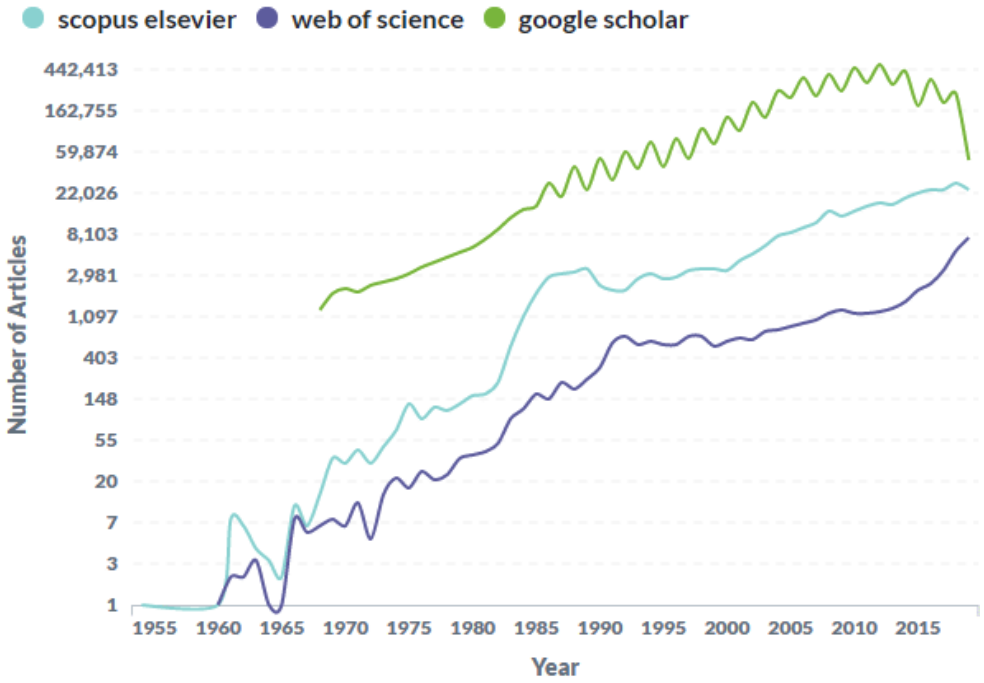

Figura 3.4: Publicaciones sobre inteligencia artificial a lo largo de los años [3].

En la actualidad, la inteligencia artificial está integrada en todas partes, proporcionando un conjunto de herramientas que mejoran la calidad de vida de las personas que utilizan esta tecnología. En la figura 3.4 se muestra un gráfico del aumento de las publicaciones relacionadas con la inteligencia artificial desde los años 50.

Entre las aplicaciones de la inteligencia artificial [33], se encuentran:

- **Hospitales y medicina**: Dentro del campo de las ciencias de la salud, la inteligencia artificial es usada como sistema de apoyo en el diagnóstico médico.
- **Servicios de atención al cliente**: La inteligencia artificial se utiliza en asistentes automatizados en línea gracias a tecnologías como el procesamiento del lenguaje natural, permitiendo a las empresas una reducción de costos de operación y capacitación [34].
- **Videojuegos**: En este campo, la inteligencia artificial es ampliamente usada para la creación de oponentes que actúan como humanos.
- **Música**: En la música, los sistemas de inteligencia artificial se centran en diversas áreas: composición, interpretación, teoría musical y procesamiento del sonido, entre otras.

Dentro del ámbito de este trabajo, *Amazon Web Services* (AWS) propone una tecnología basada en inteligencia artificial dentro de la Fórmula 1, que permite mejorar el desarrollo del vehículo en un 70 % y reducir la pérdida de carga aerodinámica de un 50 % a un 15 % [35].

La inteligencia artificial es un campo muy amplio, y es por ello que se divide en diversas ramas más específicas:

- **Robótica**: La robótica es una rama de la tecnología que estudia el diseño y construcción de máquinas capaces de desempeñar tareas realizadas por el ser humano o que requieren del uso de inteligencia.
- **Sistemas de visión**: La visión artificial (también conocida como visión por computador) es una rama de la inteligencia artificial que permite programar un computador para que reconozca una escena o las características de una imagen.

- ***Machine Learning*** o aprendizaje automático, tiene como objetivo desarrollar técnicas que permitan a las computadoras aprender a partir de información no estructurada.

- ***Deep Learning***: Es un subconjunto del aprendizaje automático centrado en el desarrollo de redes neuronales, sistemas de aprendizaje basados en el funcionamiento del cerebro humano.

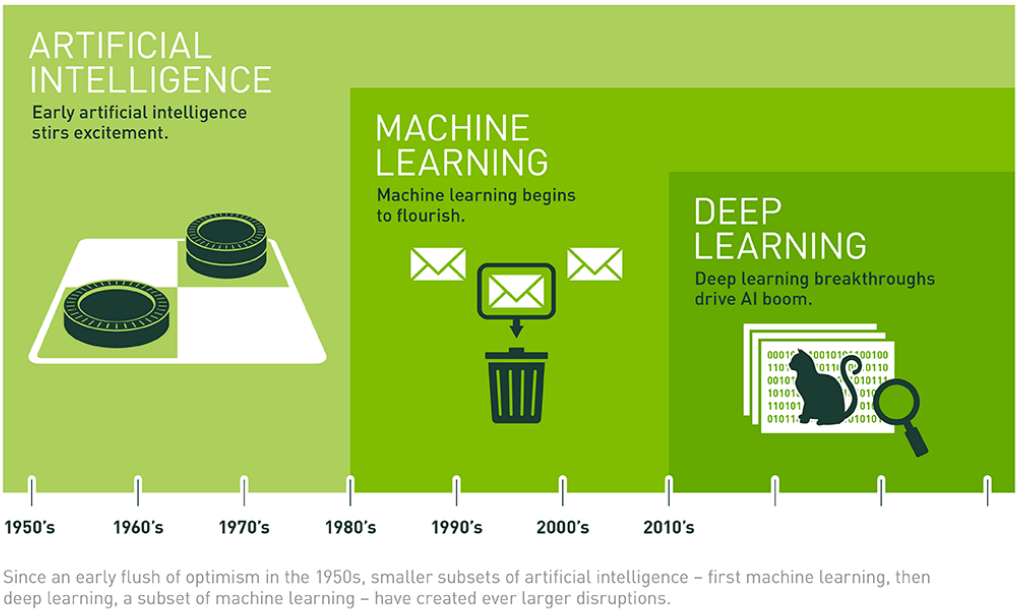

Figura 3.5: Relación entre inteligencia artificial, aprendizaje automático y aprendizaje profundo [4].

3.4. *Machine Learning*

Machine Learning o aprendizaje automático es el campo de estudio cuyo objetivo es el desarrollo de técnicas que permiten que una computadora sea capaz de aprender a partir de un conjunto de datos sin ser programada explícitamente para ello.

El objetivo principal del aprendizaje automático es la creación de un modelo que permita dar solución a un problema determinado. Estos tipos de modelos son entrenados a partir de distintos conjuntos de datos, permitiendo realizar predicciones de forma autónoma.

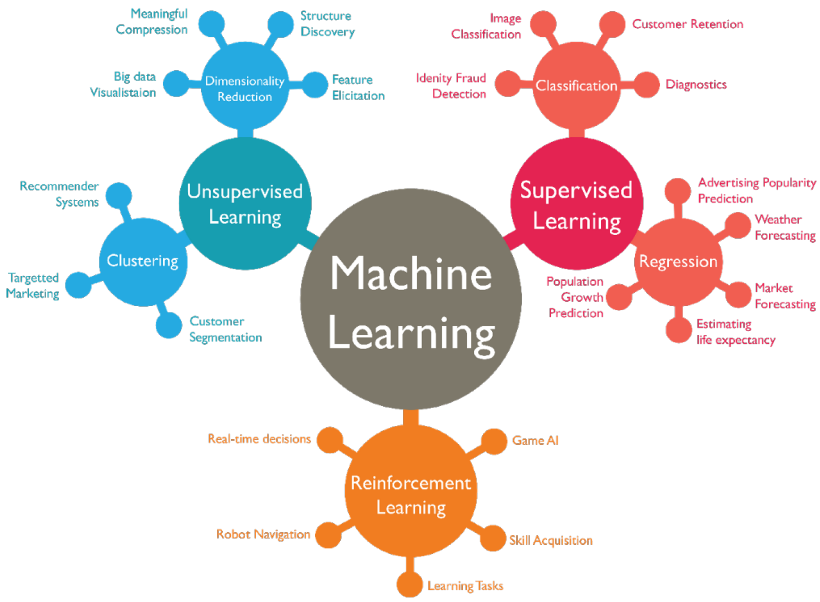

Figura 3.6: Tipos de algoritmos de aprendizaje automático [5].

Como se muestra en la figura 3.6, el aprendizaje automático se divide en distintos tipos de algoritmos, agrupados en tres grandes categorías: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo.

3.4.1. Aprendizaje supervisado

El aprendizaje supervisado es una técnica que permite entrenar un modelo en función de un conjunto de datos, denominado datos de entrenamiento. Los datos de entrenamiento están formados de una serie de ejemplos de entrenamiento que presentan una o más entradas (denominadas características) y las salidas deseadas (denominadas etiquetas).

El funcionamiento de este tipo de algoritmos se basa en poder relacionar las características con las etiquetas para obtener una predicción. Un algoritmo se dice que aprende a realizar una tarea cuando mejora su precisión de predicción con el tiempo.

Los algoritmos de aprendizaje supervisado se dividen en dos grandes grupos: algoritmos de clasificación y algoritmos de regresión.

3.4.1.1. Algoritmos de clasificación

Los algoritmos de clasificación permiten agrupar las características de un conjunto de datos en diferentes clases o etiquetas a través de reglas de decisión que permitan determinar la clase o etiqueta de un nuevo objeto dentro de una de las ya existentes.

Entre los algoritmos de clasificación se encuentran los árboles de decisión, el clasificador bayesiano o las redes neuronales artificiales. Sobre este último tipo de algoritmos se profundizará en la sección 3.5.

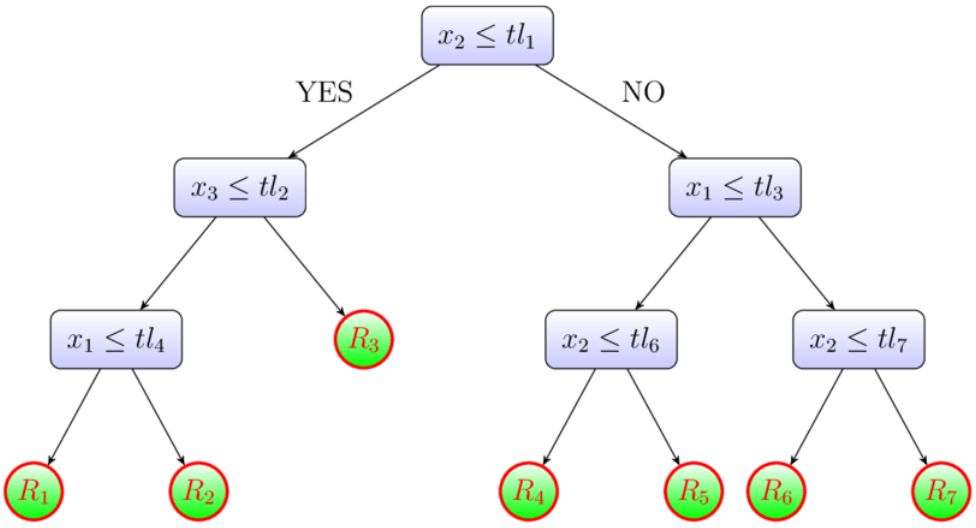

Figura 3.7: Esquema de un árbol de decisión [6].

3.4.1.2. Algoritmos de regresión

A diferencia de los algoritmos de clasificación, los algoritmos de regresión permiten ordenar un conjunto de datos entre un rango de valores. Generalmente, estos tipos de algoritmos se representan a través de una línea que separa las características del conjunto de datos [36] para tratar de establecer un patrón que describa la relación existente.

Entre los algoritmos de regresión, se encuentran la regresión lineal, la regresión de vectores de soporte (SVR) o la regresión múltiple.

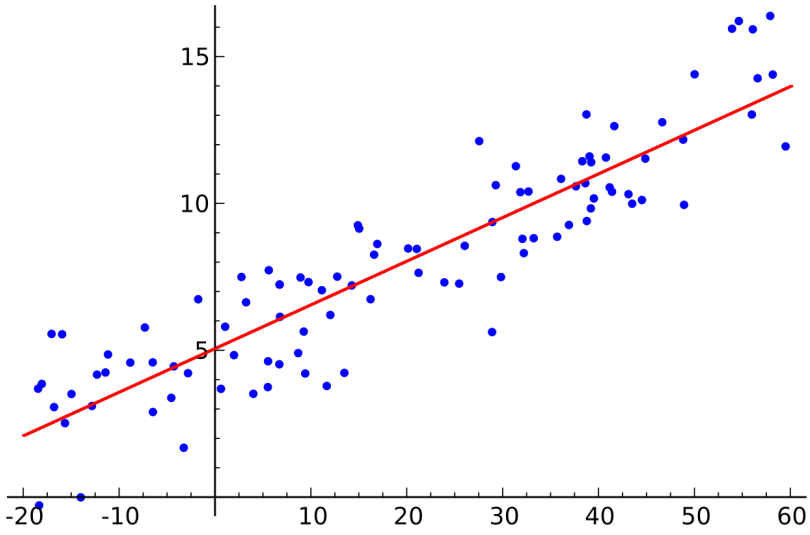

Figura 3.8: Ejemplo de regresión lineal de un conjunto de datos [7].

3.4.2. Aprendizaje no supervisado

Los algoritmos de aprendizaje no supervisado analizan un conjunto de datos no etiquetados (que solo contiene entradas o características) e intentan encontrar una estructura en los datos. Por lo tanto, estos tipos de algoritmos aprenden a partir de datos que no han sido clasificados o categorizados previamente.

El objetivo principal del aprendizaje no supervisado es identificar puntos en común entre los datos y reaccionar en base a estos puntos por cada nuevo dato del conjunto de datos [37].

Dentro del aprendizaje no supervisado, se encuentran dos grandes grupos: los algoritmos de reducción de dimensión y los algoritmos de *clustering*.

3.4.2.1. Algoritmos de reducción de dimensión

La reducción de dimensionalidad tiene como objetivo reducir el número de variables aleatorias que se tiene en consideración en un conjunto de datos a la hora de realizar el análisis de los datos.

Uno de los algoritmos más populares de reducción de dimensión es el denominado *Principal Component Analysis* (PCA) [38]. Este algoritmo permite proyectar solo los primeros componentes principales de un conjunto de datos para obtener datos de menor dimensión y preservar la mayor cantidad posible de variación de los mismos. Es bastante útil cuando se quiere representar un conjunto de datos con múltiples características en un plano.

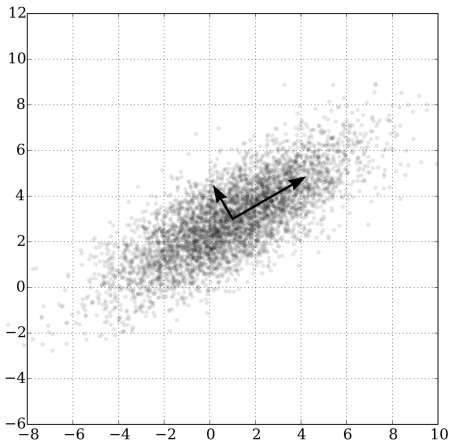

Figura 3.9: Ejemplo de regresión lineal de un conjunto de datos [8].

3.4.2.2. Algoritmos de *clustering*

Clustering o agrupamiento es la técnica de agrupar un conjunto de datos sin etiquetar en diferentes grupos (también conocidos como clusters). Los grupos están formados con datos que presentan características similares entre ellos. Por lo general, se distinguen dos tipos de agrupamiento:

- ***Hard clustering*** o agrupamiento duro: En este tipo de agrupamiento, cada dato debe pertenecer a un grupo concreto.
- ***Soft clustering*** o agrupamiento blando: A diferencia del agrupamiento duro, cada dato puede pertenecer a un grupo hasta cierto punto, llegando a ser dudosa su pertenencia a más de un grupo.

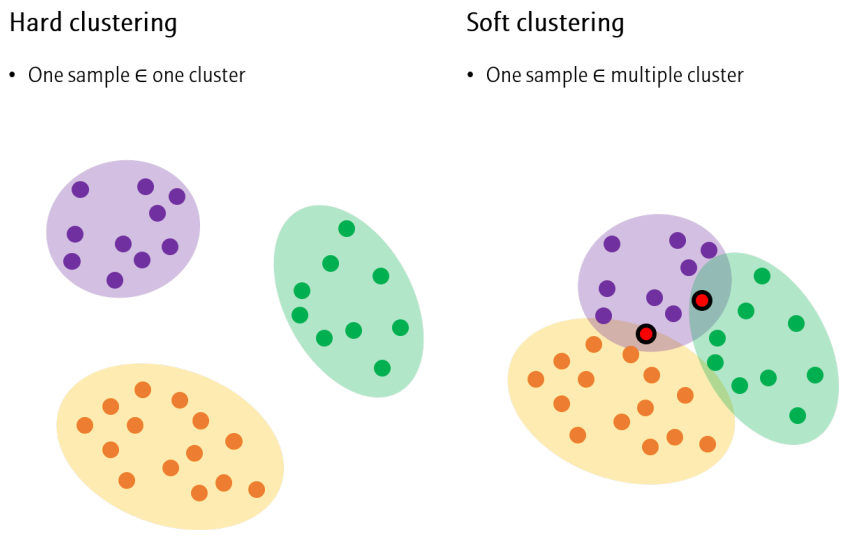

Figura 3.10: Ejemplo de agrupamiento duro y agrupamiento blando [9].

Actualmente, existen multitud de algoritmos de agrupamiento que se dividen por modelos: modelo de densidad, modelo de conectividad o modelo de centroides, entre otros. Este último modelo es uno de los más populares debido a la simplicidad de sus algoritmos, como el algoritmo de *K-Means* [39].

El algoritmo de *K-Means* es un algoritmo iterativo, como se observa en la figura 3.11, donde se definen k números fijos de centroides que identifican la ubicación de k grupos. En cada iteración, el algoritmo asigna cada dato del conjunto de datos al grupo con el centroide más cercano mediante el cálculo de la distancia media cuadrática.

Para obtener el número óptimo de grupos dentro del algoritmo de *K-Means* [40], se utilizan varios métodos y medidas:

- ***Elbow Method* o Codo de Jambú**: Este método consiste en calcular diferentes cantidades de grupos e ir comprobando la similitud de los datos dentro de los distintos grupos.
- **Coeficiente de silueta**: Es una medida que indica cómo de compactos y separados se encuentran los diferentes grupos. Por lo tanto, el número de grupos óptimo será aquel que tenga un mayor coeficiente de silueta.

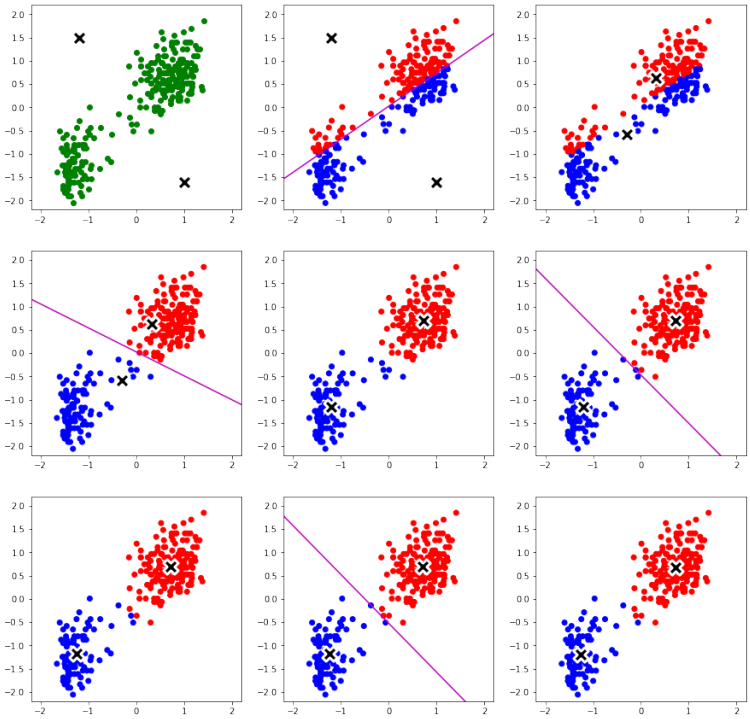

Figura 3.11: Funcionamiento del algoritmo de *K-Means* en cada iteración [10].

3.4.3. Aprendizaje por refuerzo

El aprendizaje por refuerzo es un método de aprendizaje automático que se basa en recompensar los comportamientos deseados y penalizar los no deseados. Dentro de este contexto, el sistema de aprendizaje (también conocido como agente) deberá aprender por él mismo cuál es la mejor estrategia (también conocida como política) que define la acción que el agente deberá elegir en una situación específica para obtener el mayor número de recompensas por tiempo [41].

Este tipo de aprendizaje es utilizado en el mundo de los videojuegos para lograr que entidades o personajes aprendan a realizar las mejores decisiones en base a criterios específicos.

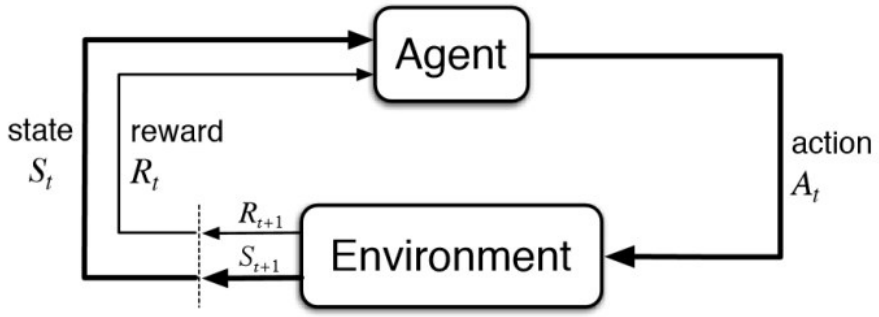

Figura 3.12: Esquema de funcionamiento del aprendizaje por refuerzo [11].

3.5. *Deep Learning*

Deep Learning o aprendizaje profundo es un subconjunto del aprendizaje automático basado en estructuras lógicas que simulan el comportamiento del cerebro humano: las redes neuronales artificiales.

Las redes neuronales artificiales son estructuras formadas por representaciones sucesivas de un conjunto de datos, denominadas capas. Cada una de las capas

de una red neuronal está formada por una serie de unidades llamadas neuronas. En la figura 3.13 se puede apreciar el procesamiento que se realiza dentro de una neurona de una red neuronal artificial.

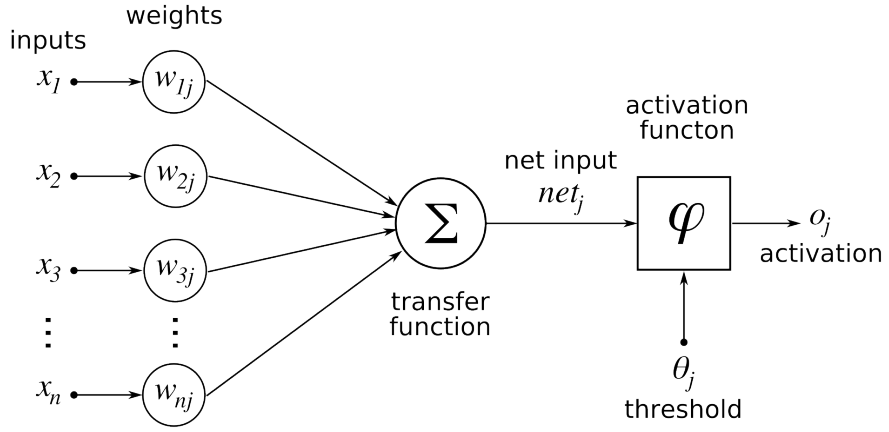

Figura 3.13: Esquema de funcionamiento del procesamiento de una neurona [12].

En primer lugar, la neurona recibe una serie de datos de entrada, que son multiplicados cada uno de ellos por una serie de pesos. A continuación, los valores son sumados y, sobre el resultado de dicha suma, se aplica una función de activación para generar el resultado de salida de la neurona.

3.5.1. Estructura de una red neuronal artificial

Una red neuronal artificial está formada por una serie de neuronas agrupadas en capas. Existen 3 tipos de capas en una red neuronal artificial:

- **Capa de entrada**: Esta capa contiene las neuronas que reciben los datos de entrada de un conjunto de datos.
- **Capa oculta**: Contiene las neuronas internas de la red: reciben datos de neuronas de la capa anterior, aplican un procesamiento y generan una salida parcial.

- **Capa de salida**: Es la última capa de la red que genera el resultado obtenido al procesar todo el conjunto de datos.

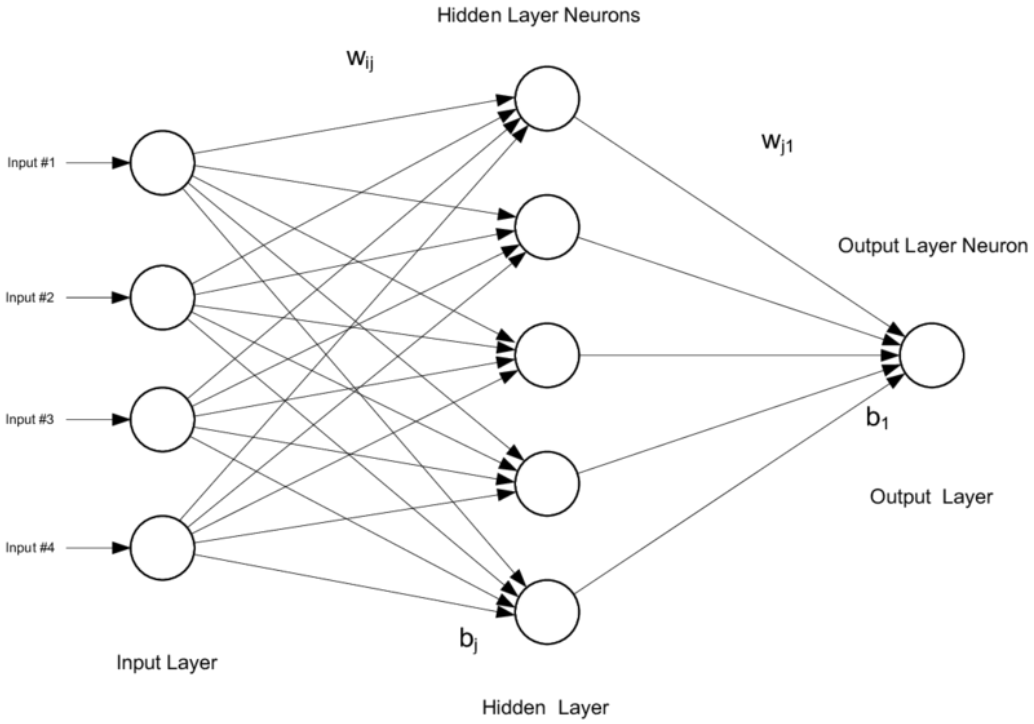

Figura 3.14: Estructura de una red neuronal artificial [13].

Debido a esta estructura formada por capas, es posible aplicar transformaciones diferentes en cada capa y sobre cada dato, permitiendo extraer automáticamente características o detalles cada vez más precisos y concretos.

3.5.2. Entrenamiento de una red neuronal artificial

Una red neuronal artificial debe de entrenar para generar un resultado. El entrenamiento es el proceso que modifica el valor de los pesos de la red neuronal artificial asociados a cada neurona con el fin de que pueda generar una salida a partir de un conjunto de datos de entrada.

Para que una red neuronal artificial aprenda y mejore su salida, se aplica el algoritmo de *backpropagation*. Este algoritmo calcula el descenso del gradiente de una función de error (normalmente el error cuadrático medio) con respecto a los pesos de la red neuronal artificial [42].

3.5.3. Optimización de hiperparámetros

El principal objetivo de una red neuronal es predecir correctamente la salida o el resultado esperado sobre un conjunto de datos. Para lograr este objetivo, es necesario determinar el valor que debe tener una serie de hiperparámetros que se utilizan para aplicar el procesamiento sobre cada una de las neuronas de la red neuronal artificial.

Los hiperparámetros son los parámetros que se definen antes de entrenar un modelo específico de red neuronal: número de capas, funciones de activación, número de iteraciones, etc. La optimización de estos hiperparámetros es un paso muy importante que se debe realizar en cualquier modelo de aprendizaje profundo, ya que afecta de forma directa al rendimiento del mismo.

En la actualidad, existen paquetes y bibliotecas como KerasTuner [43], que permiten realizar una optimización de hiperparámetros de forma sencilla a través de diferentes algoritmos de búsqueda.

4. Metodología

Durante la elaboración de este trabajo se ha seguido una metodología ágil, en concreto la metodología SCRUM [44]. La metodología SCRUM es una metodología iterativa e incremental donde los requisitos evolucionan a lo largo del desarrollo. Esta metodología presenta tres fases principales:

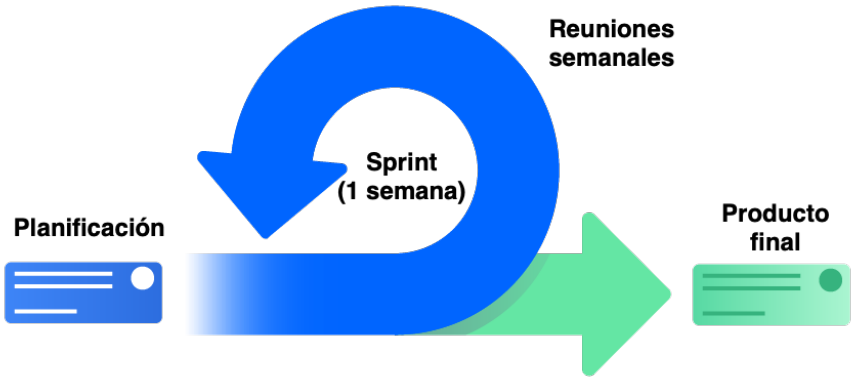

Figura 4.1: Metodología SCRUM aplicada en el proyecto.

- **Planificación**: Durante la planificación se desarrolla la visión general del trabajo, dando únicamente detalles de las fases o funcionalidades principales del proyecto.
- **Sprint** (iteración): Son intervalos de tiempo en donde se desarrolla las distintas fases o funcionalidades del proyecto de forma parcial. Con cada iteración, el proyecto incrementa las fases o funcionalidades definidas en la planificación.

- **Control**: Después de cada sprint, se realiza una fase de control que evalúa y mide el progreso alcanzado durante el sprint, gracias a la retroalimentación del tutor. Además, se definen los objetivos de la siguiente iteración.

4.1. Planificación del trabajo

La planificación de este proyecto se compone de cinco tareas o fases principales que van desde labores de búsqueda de información hasta la documentación de los resultados obtenidos. Las principales tareas que se han realizado durante la realización de este trabajo son:

- **Búsqueda de información**: En esta fase se definen los objetivos del proyecto y se comienza la búsqueda de información relacionada con el mismo.
- **Investigación de herramientas**: Se aprenden, analizan y prueban distintas herramientas útiles para la realización del proyecto.
- **Recogida de datos**: Durante esta fase, se recogen, procesan y limpian los datos utilizados en el análisis e implementación del proyecto.
- **Implementación**: Se desarrollan, entrenan y evalúan los modelos y aplicaciones desarrolladas durante el proyecto.
- **Documentación**: Finalmente, se realiza una memoria que resume todo el trabajo realizado.

La planificación inicial de este trabajo consta de una duración de poco más de cuatro meses, desde febrero hasta principios de junio. En las figuras 4.2 y 4.3 se muestra el diagrama de Gantt y la duración de cada tarea definida, respectivamente.

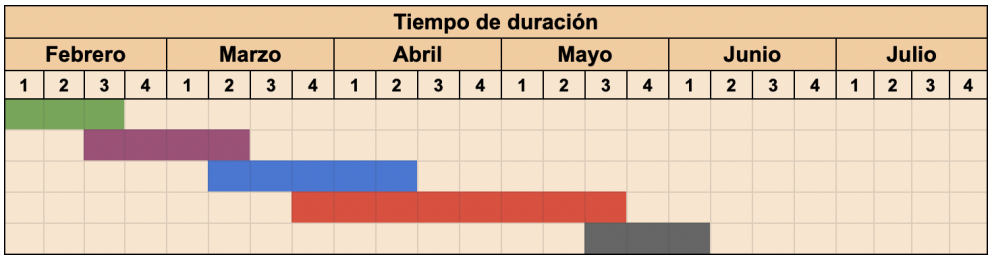

Figura 4.2: Diagrama de Gantt de la planificación inicial del trabajo.

Tareas	Tiempo estimado (h)	Inicio	Final
Búsqueda de información	40	01/02/2022	20/02/2022
Investigación de herramientas	60	14/02/2022	13/03/2022
Recogida de datos	50	07/03/2022	13/04/2022
Implementación	110	21/03/2022	29/05/2022
Documentación	40	23/05/2022	12/06/2022
Horas totales	300		

Figura 4.3: Fechas y duración de las tareas de la planificación inicial.

Sin embargo, debido a varias circunstancias no se ha podido cumplir la planificación inicial y se ha alargado la duración del trabajo. En las figuras 4.4 y 4.5 se observa el diagrama de Gantt y la duración de cada tarea definida, respectivamente.

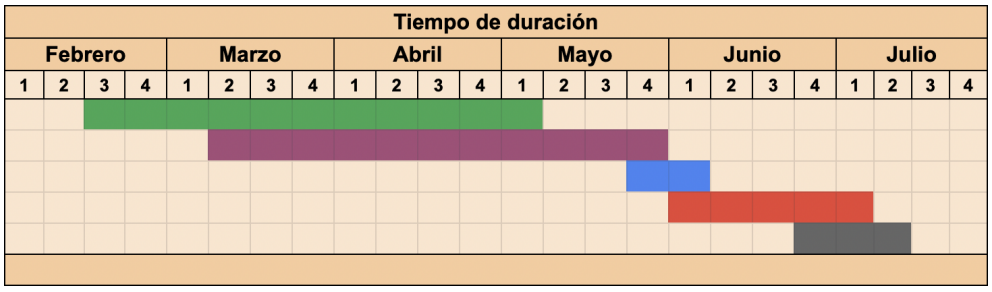

Figura 4.4: Diagrama de Gantt de la planificación final del trabajo.

Tareas	Tiempo estimado (h)	Inicio	Final
Búsqueda de información	40	14/02/2022	08/05/2022
Investigación de herramientas	100	07/03/2022	29/05/2022
Recogida de datos	30	23/05/2022	05/06/2022
Implementación	150	30/05/2022	03/07/2022
Documentación	40	27/06/2022	10/07/2022
Horas totales	360		

Figura 4.5: Fechas y duración de las tareas de la planificación final.

Como se puede comparar con la planificación inicial, las tareas de búsqueda de información e investigación de herramientas se han alargado a lo largo del tiempo. A diferencia de la planificación inicial que se había estimado una duración de 300 horas, el desarrollo final del trabajo se ha alargado hasta las 360 horas.

4.2. Sistema de control de versiones

Durante el desarrollo de la aplicación web que muestra la visualización de los datos de Fórmula 1 es recomendable mantener una gestión de la configuración. La gestión de la configuración permite mantener un estricto control de todos los cambios realizados sobre el producto en desarrollo [45].

Para ello, se ha utilizado la herramienta Git [46], un sistema de control de versiones distribuido, para crear un repositorio remoto a través de GitHub para gestionar el código fuente del proyecto [47]. El flujo de trabajo utilizado se denomina Gitflow [48], que define un estricto modelo de ramas diseñado alrededor de las funcionalidades del proyecto.

Figura 4.6: Logo de la herramienta Git [14].

Para la gestión de la configuración se han seguido las buenas prácticas de integración continua aprendidas durante el grado: desarrollar funcionalidades pequeñas, implementar cambios frecuentes o realizar mensajes detallados en cada *commit* del código, entre otras prácticas.

5. Tecnologías y herramientas

En este apartado se resumen las tecnologías y herramientas utilizadas durante la implementación y el desarrollo del trabajo.

5.1. Lenguajes de programación

Para la realización de este proyecto se han utilizado dos lenguajes de programación: Python, para la extracción y análisis de los datos de Fórmula 1; y JavaScript, para la creación de la aplicación web.

5.1.1. Python

Python [49] es un lenguaje de programación de propósito general, normalmente usado para el análisis de datos. Su versatilidad y facilidad de uso lo hace uno de los lenguajes de programación más populares hoy en día, como se puede observar en la figura 5.1. Debido a la activa comunidad alrededor de este lenguaje, Python se ha convertido en el lenguaje de programación líder dentro del mundo de la investigación de inteligencia artificial e ingeniería de datos.

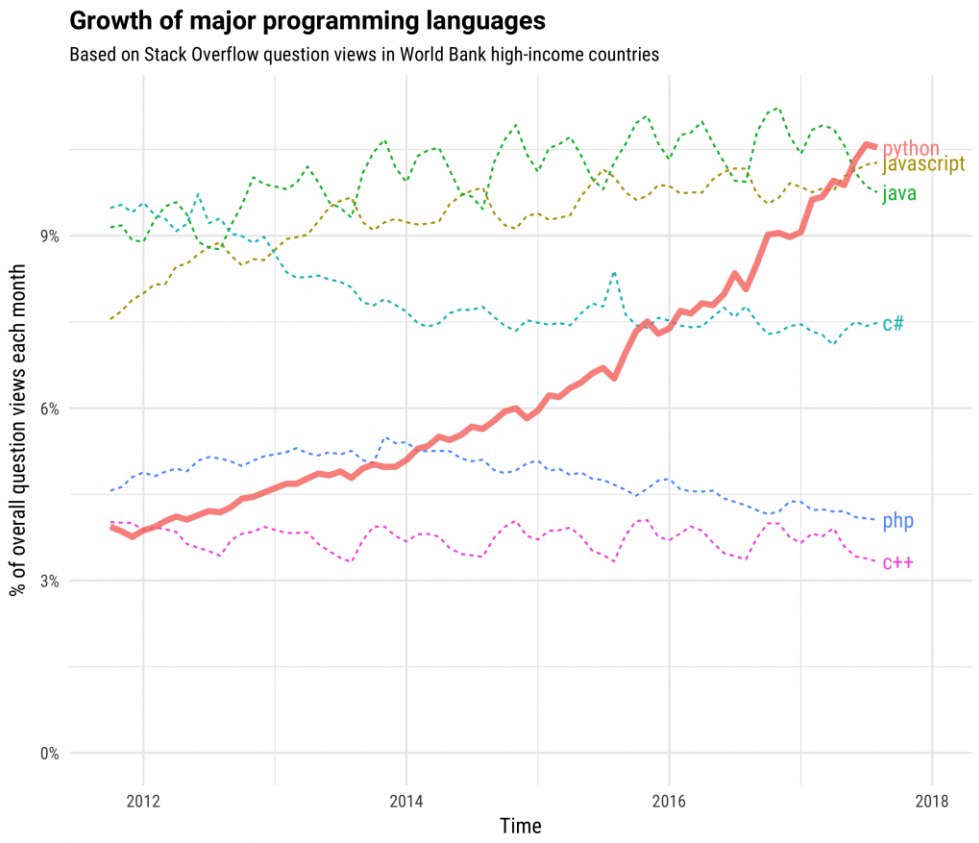

Figura 5.1: Uso de Python comparado con otros lenguajes de programación [15].

El desarrollo de este proyecto se ha realizado con este lenguaje de programación debido a dos motivos principales:

- La gran abundancia de bibliotecas, paquetes y ayudas dentro del campo de la inteligencia artificial e ingeniería de datos, que han permitido un desarrollo rápido y eficiente del análisis de los datos.
- Por otro lado, la API que proporcionaba los datos de Fórmula 1 está implementada en Python, y por lo tanto la extracción de los mismos se ha realizado en este lenguaje.

5.1.2. JavaScript

JavaScript [50] es un lenguaje de programación de alto nivel utilizado principalmente para añadir características interactivas dentro de una página web. Es una de las tecnologías principales de la web, junto con HTML y CSS, aunque se puede extrapolar a cualquier máquina gracias a tecnologías como Node.js [51].

Figura 5.2: Logo oficial de JavaScript [16].

Se ha utilizado este lenguaje en el desarrollo de este proyecto para generar la aplicación web mediante diversas tecnologías, como React, FastAPI y ChartJS [24, 25, 52], que permiten visualizar los datos de Fórmula 1 de forma dinámica e interactiva.

5.2. Entornos de desarrollo

Para la recolección y análisis de los datos de Fórmula 1 se ha utilizado el entorno de Jupyter Notebooks, mientras que para la creación de la aplicación web se ha utilizado el editor de texto Visual Studio Code. Con respecto a la realización de la documentación, se ha utilizado Overleaf como plataforma para crear documentos con LaTeX.

5.2.1. Jupyter Notebook

Jupyter Notebook [53] es un entorno de desarrollo de código abierto que permite crear y compartir documentos, visualizaciones, texto explicativo y código en vivo. Este entorno está estructurado en forma de celdas ejecutables por separado, que facilitan la documentación del código y la visualización de los datos.

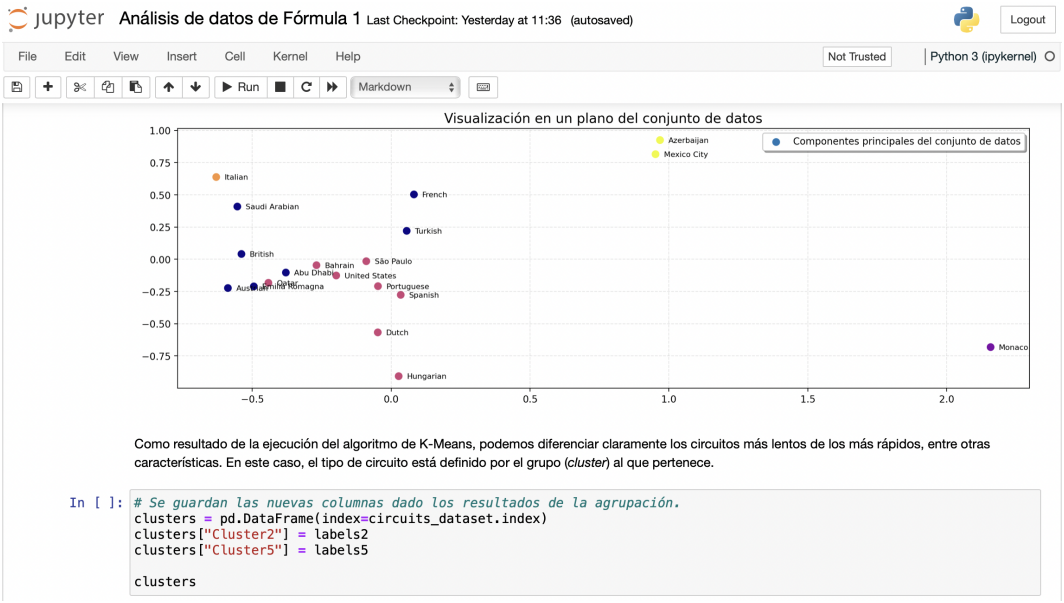

Figura 5.3: Ejemplo de la interfaz de usuario de Jupyter Notebook.

Jupyter Notebook se ejecuta como una aplicación web como se puede observar en la figura 5.3 y, por lo tanto, no es necesario ninguna configuración adicional para comenzar a programar.

5.2.2. Visual Studio Code

Visual Studio Code (también conocido como VSCode) [54] es un editor de texto desarrollado por Microsoft. Su amplia comunidad y las numerosas exten-

siones existentes lo hacen el editor de texto más usado entre programadores y desarrolladores de todo tipo.

Figura 5.4: Logo oficial de Visual Studio Code [17].

Se ha elegido este entorno sobre otros para desarrollar la aplicación web por dos motivos principales:

- Debido al gran número de extensiones existentes en Visual Studio Code, el tiempo de trabajo sobre la aplicación web ha disminuido considerablemente.
- La herramienta Git está integrada por defecto dentro del entorno de desarrollo, lo que ha permitido una mejor gestión de la configuración del proyecto.

5.2.3. Overleaf

Overleaf [55] es una plataforma en la nube que permite gestionar y compartir proyectos de LaTeX con múltiples usuarios. Gracias a su facilidad de uso y su amplia documentación, se ha convertido en la plataforma líder para crear documentos de este tipo.

LaTeX [56] es un sistema de composición de textos, orientado a la creación de documentos escritos que presenten una alta calidad tipográfica. Es ampliamente usado en tesis e investigaciones científicas.

5.3. Tecnologías externas

A continuación se detallan las tecnologías, herramientas, paquetes y/o bibliotecas que se han utilizado durante el desarrollo del trabajo.

5.3.1. FastF1

FastF1 [57] es una API no oficial que recoge datos históricos y de telemetría de la Fórmula 1. Está desarrollada en Python y construida sobre el paquete Pandas [58]. A través de esta API, se puede acceder a tiempos de vuelta, telemetría y posición del vehículo, datos meteorológicos y resultados de una sesión, entre otros muchos datos.

5.3.2. Pandas

Pandas [58] es un paquete de código abierto desarrollado en Python que ofrece estructuras de datos flexibles para facilitar la manipulación y el tratamiento de los datos. Durante la realización de este proyecto se han utilizado los *dataframes*, estructuras bidimensionales (tablas) donde todos los datos de una misma columna son del mismo tipo y las filas son registros que pueden contener datos de distintos tipos.

Figura 5.5: Logo oficial de Pandas [18].

5.3.3. Matplotlib

Matplotlib [59] es una biblioteca de Python para realizar visualizaciones de datos en dos dimensiones. Esta biblioteca está diseñada para realizar gráficos de forma simple con pocos comandos.

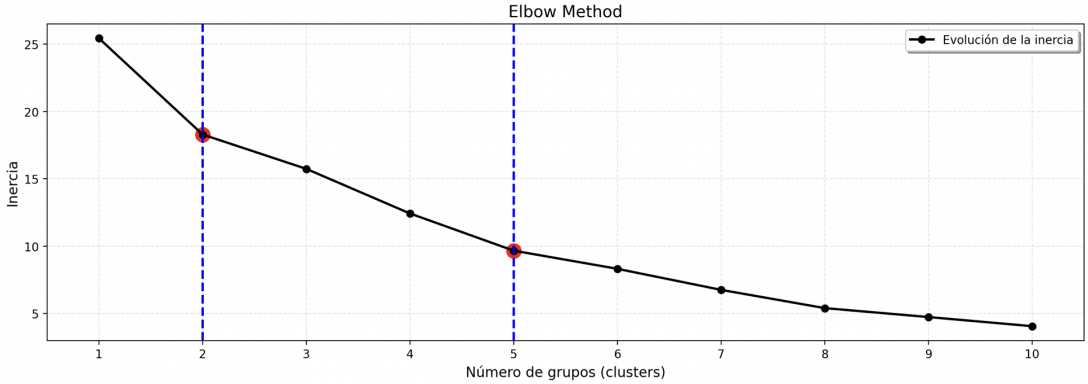

Figura 5.6: Ejemplo de visualización usando la biblioteca Matplotlib.

Como se muestra en la figura 5.6, esta biblioteca ha sido de gran ayuda para realizar visualizaciones y análisis de los datos de Fórmula 1 durante este trabajo.

5.3.4. TensorFlow & Keras

TensorFlow [60] es una biblioteca de código abierto desarrollada por Google para el aprendizaje automático. TensorFlow puede entrenar y ejecutar redes neuronales profundas para la clasificación de dígitos escritos a mano, el reconocimiento de imágenes, los modelos de secuencia para la traducción automática o el procesamiento del lenguaje natural, entre otras tareas.

Figura 5.7: Logo oficial de TensorFlow [19].

Por otro lado, Keras [61] es una biblioteca de redes neuronales construida dentro de TensorFlow. Keras ofrece una API de alto nivel para la creación de modelos de aprendizaje profundo a través de una interfaz sencilla y una sintaxis homogénea, modular y ampliable.

5.3.5. KerasTuner

KerasTuner [43] es un *framework* escalable de optimización de hiperparámetros que permite resolver los problemas de búsqueda de hiperparámetros. Esta biblioteca presenta varios algoritmos de búsqueda como la optimización bayesiana, aunque también permite experimentar con nuevos algoritmos de búsqueda.

Para la realización de este trabajo, se ha utilizado esta biblioteca para disminuir el tiempo de desarrollo a la hora de optimizar las redes neuronales artificiales construidas.

5.3.6. Scikit Learn

Scikit Learn [62] es una biblioteca de código abierto desarrollada en Python que se utiliza para implementar algoritmos de Machine Learning. Scikit Learn proporciona funcionalidades que ayudan a crear modelos de clasificación, regresión y agrupación, entre otros tipos de algoritmos.

Figura 5.8: Logo oficial de Scikit Learn [20].

5.3.7. Node.js

Node.js [51] es un entorno de tiempo de ejecución de JavaScript. Este entorno de tiempo de ejecución en tiempo real incluye todo lo que se necesita para ejecutar un programa escrito en JavaScript.

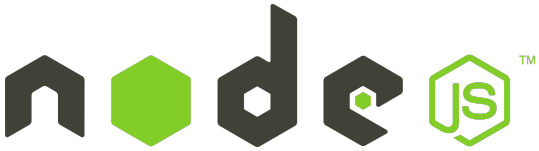

Figura 5.9: Logo oficial de Node.js [21].

Esta herramienta integra el motor V8 de Google para ejecutar bibliotecas y paquetes escritos en JavaScript en cualquier máquina. Se ha utilizado esta herramienta en el proyecto para desarrollar la aplicación web.

5.3.8. React

React es un *framework* de JavaScript focalizado en el desarrollo de interfaces de usuario. React permite el desarrollo de aplicaciones web de una sola página (SPA) a través de componentes reutilizables. Fue creada por Facebook y es el *framework* de JavaScript más utilizado en todo el mundo para crear aplicaciones web.

Figura 5.10: Logo oficial de React [22].

5.3.9. FastAPI

FastAPI [25] es un *framework* de alto rendimiento desarrollado en Python para construir APIs. Se ha utilizado este *framework* para recopilar la información de Fórmula 1 (a través de Python) y visualizar todos los datos recogidos en la aplicación web (a través de JavaScript).

5.3.10. Chart.js

Chart.js [52] es una biblioteca de JavaScript de código abierto para la visualización de datos. Durante el desarrollo de la aplicación web, se ha utilizado esta biblioteca para mostrar los datos de telemetría de la Fórmula 1. En la figura 5.11 se puede observar un ejemplo de visualización de datos con este paquete.

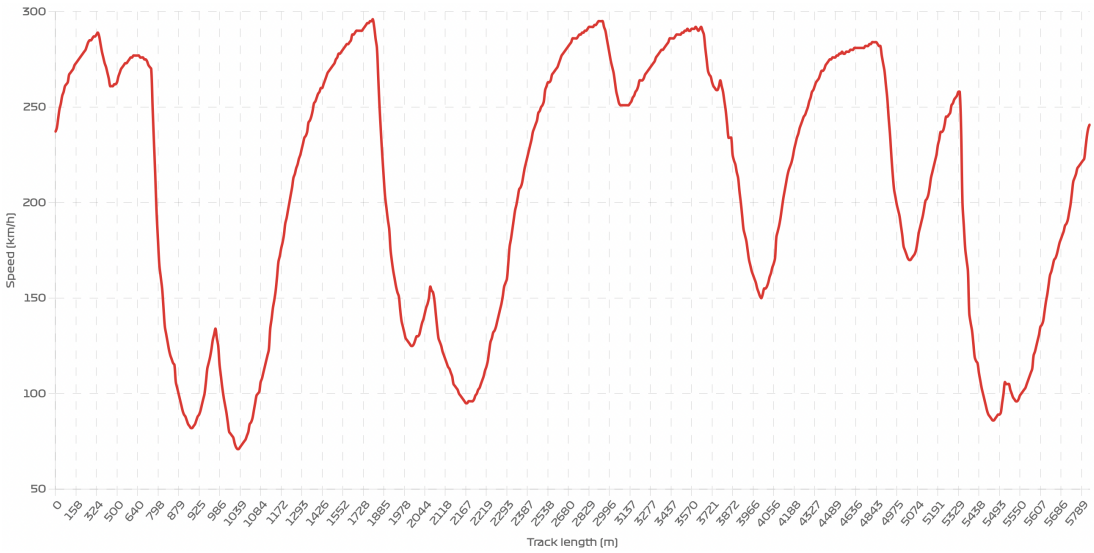

Figura 5.11: Ejemplo de visualización usando la biblioteca Chart.js [23].

6. Implementación y desarrollo

En este apartado se explica todo el proceso de implementación y desarrollo del proyecto: obtención y procesamiento de los datos de Fórmula 1, análisis y evaluación de los modelos de aprendizaje automático construidos y visualización de la telemetría de Fórmula 1 a través de una aplicación web.

En la obtención y posterior análisis de los datos, se ha decidido únicamente utilizar aquellos datos de la temporada 2021 de Fórmula 1, debido a la multitud de diferencias existentes entre varias temporadas: cambios de pilotos entre escuderías o llegadas de nuevos pilotos, distintos circuitos en el calendario, vehículos con nuevas piezas respecto al año anterior, etc.

6.1. Obtención y procesamiento de los datos

La obtención y el procesamiento de los datos de Fórmula 1 se ha realizado a través de la biblioteca FastF1 [57]. Para el análisis y evaluación de los modelos de aprendizaje automático se han necesitado dos conjuntos de datos diferentes:

- **Conjunto de datos de circuitos**: Este conjunto de datos reúne la información de los diferentes circuitos de la temporada 2021.
- **Conjunto de datos de pilotos**, que agrupa los resultados individuales de cada piloto durante la temporada 2021.

Adicionalmente, se ha creado un archivo JSON [63] auxiliar que guarda identificadores únicos de pilotos, escuderías, circuitos y tipos de compuestos del neumático, para poder evaluar los modelos de aprendizaje automático de forma más sencilla.

6.1.1. Conjunto de datos de circuitos

Este conjunto de datos presenta los datos de todos los circuitos del calendario oficial de la Fórmula 1 durante el año 2021. Para la obtención de esta información, se ha tomado en cuenta la vuelta más rápida durante la clasificación en cada circuito. Los datos pertenecientes al Gran Premio de Estiria, al Gran Premio de Bélgica y al Gran Premio de Rusia no se han tenido en cuenta por los siguientes motivos:

- El Gran Premio de Estiria se realizó en el mismo circuito que el Gran Premio de Austria y, por lo tanto, se ha excluido para evitar duplicaciones.
- El Gran Premio de Bélgica no se disputó debido a la fuerte lluvia. Es por ello que no se tendrá en cuenta en este análisis.
- La lluvia durante la clasificación altera los resultados de los pilotos ya que el rendimiento del vehículo se iguala prácticamente entre todas las escuderías. Por lo tanto, debido a las condiciones de lluvia del Gran Premio de Rusia, se excluye este circuito del conjunto de datos.

En la figura 6.1 se muestra parte del conjunto de datos resultante tras el procesamiento de los datos. Este conjunto de datos presenta un total de 19 columnas diferentes:

- ***GrandPrix***: Nombre del circuito.

- ***AvgSpeed***: Velocidad media por vuelta del circuito.
- ***DistanceDRS***: Distancia total de DRS [64] en una vuelta del circuito.
- ***Distance***: Distancia máxima del circuito.
- ***MaxSpeed***: Velocidad máxima durante una vuelta del circuito.
- ***MinSpeed***: Velocidad mínima durante una vuelta del circuito.
- ***MaxThrottle***: Porcentaje de máxima aceleración durante una vuelta del circuito.
- El resto de columnas indican el número de rectas y curvas del circuitos categorizadas por velocidades. Las rectas se agrupan por su velocidad máxima y las curvas se agrupan por su velocidad mínima.

GrandPrix	AvgSpeed	DistanceDRS	Distance	MaxSpeed	MinSpeed	MaxThrottle	C50	C100	C150	C200	C250	S250	S300	S150	S200	S50	S100
Bahrain Grand Prix	218.342318	1667.035000	5391.107500	315.0	74.0	0.636119	1.0	3.0	3.0	1.0	0.0	4.0	4.0	0.0	0.0	0.0	0.0
Emilia Romagna Grand Prix	237.668831	913.927778	4899.232778	322.0	111.0	0.727273	0.0	2.0	4.0	3.0	0.0	1.0	6.0	1.0	1.0	0.0	0.0
Portuguese Grand Prix	212.910494	918.382500	4627.868611	314.0	77.0	0.611111	0.0	5.0	1.0	1.0	0.0	3.0	3.0	0.0	1.0	0.0	0.0
Spanish Grand Prix	219.068966	1189.658611	4665.113611	316.0	89.0	0.633229	0.0	3.0	5.0	0.0	0.0	2.0	3.0	1.0	2.0	0.0	0.0
Monaco Grand Prix	168.225256	278.981111	3276.551111	287.0	47.0	0.450512	5.0	5.0	4.0	2.0	0.0	3.0	2.0	3.0	4.0	2.0	2.0

Figura 6.1: Primeros elementos del conjunto de datos de circuitos.

6.1.2. Conjunto de datos de pilotos

Este conjunto de datos contiene la información acerca de los resultados de los diferentes pilotos que componen el campeonato mundial de la Fórmula 1 durante la temporada 2021. Los datos de los pilotos pertenecientes al Gran Premio de Bélgica y al Gran Premio de Rusia no se ha tenido en cuenta por los siguientes motivos:

- El Gran Premio de Bélgica no se disputó debido a la fuerte lluvia y, por lo tanto, se excluye de este análisis.
- La lluvia durante la clasificación altera los resultados de los pilotos ya que el rendimiento del vehículo se iguala prácticamente entre todas las escuderías. Por lo tanto, debido a las condiciones de lluvia del Gran Premio de Rusia, se excluye este circuito del conjunto de datos.

Por otro lado, se excluyen los datos relacionados con el piloto Robert Kubica, que disputó solamente dos carreras durante la temporada debido a una sustitución propiciada por la pandemia [65]. El conjunto de datos está formado por 11 columnas, cuya información se explica a continuación:

- ***Driver***: Nombre del piloto.
- ***Team***: Nombre de la escudería a la que pertenece el piloto.
- ***GrandPrix***: Nombre del circuito.
- ***Laps***: Número de vueltas durante la carrera.
- ***Length***: Distancia de una vuelta del circuito.
- ***LapTime***: Tiempo en segundos de la mejor vuelta del piloto durante la clasificación.
- ***Compound***: Tipo de compuesto del neumático utilizado durante la clasificación.
- ***AvgSpeed***: Velocidad media del piloto durante la vuelta de clasificación.
- ***MaxSpeed*** Velocidad máxima del piloto durante la vuelta de clasificación.
- ***QualiPos***: Posición de salida del piloto en la carrera.

- ***RacePos***: Resultado del piloto en la carrera.

En la figura 6.2 se observa parte del conjunto resultante tras la obtención y procesamiento de los resultados individuales de los pilotos durante la temporada 2021.

	Driver	Team	GrandPrix	Laps	Length	LapTime	Compound	AvgSpeed	MaxSpeed	QualiPos	RacePos
0	Lewis Hamilton	Mercedes	Bahrain Grand Prix	56.0	5378.502778	89.385	SOFT	217.504021	314	2.0	1.0
1	Max Verstappen	Red Bull Racing	Bahrain Grand Prix	56.0	5378.502778	88.997	SOFT	218.342318	315	1.0	2.0
2	Valtteri Bottas	Mercedes	Bahrain Grand Prix	56.0	5378.502778	89.586	SOFT	216.873995	312	3.0	3.0
3	Lando Norris	McLaren	Bahrain Grand Prix	56.0	5378.502778	89.974	SOFT	215.943850	316	7.0	4.0
4	Sergio Perez	Red Bull Racing	Bahrain Grand Prix	56.0	5378.502778	90.659	MEDIUM	214.904000	310	0.0	5.0

Figura 6.2: Primeros elementos del conjunto de datos de pilotos.

6.1.3. Información auxiliar

Como información auxiliar se guardan los identificadores únicos de todos los pilotos, escuderías, circuitos y tipos de compuestos del neumático durante la temporada 2021 de Fórmula 1.

Estos datos auxiliares se guardan en un fichero JSON [63], ya que su estructura familiar permite un reconocimiento rápido de los distintos elementos. Este fichero será usado de ayuda para realizar la evaluación de los distintos modelos de aprendizaje automático.

6.2. Análisis y evaluación

En esta sección se explican los algoritmos y técnicas de aprendizaje automático utilizados en el análisis y evaluación de los conjuntos de datos obtenidos en el apartado anterior. Para ello, el análisis de los conjuntos de datos se dividirá en dos partes principales:

- ***Clustering* de circuitos**: Se aplica el algoritmo de *K-Means* para agrupar los distintos circuitos presentes en la temporada 2021 de Fórmula 1.
- **Predicción de resultados**: Se construyen varios modelos de redes neuronales artificiales que permiten clasificar los resultados de un piloto en un Gran Premio de Fórmula 1.

6.2.1. *Clustering* de circuitos

El conjuntos de datos utilizado para aplicar el algoritmo de agrupamiento es el obtenido en la sección 6.1.1. La normalización de los datos es un aspecto importante dentro del aprendizaje automático, ya que permite utilizar una escala común entre todo el conjunto de datos y obtener mejores resultados.

En este caso, se utiliza un escalador MinMax [66] presente dentro del paquete Scikit Learn [62]. Este escalador permite normalizar (y desnormalizar) los datos de un conjunto de datos en una escala entre 0 y 1. En la figura 6.3 se puede observar el resultado de aplicar el escalador MinMax al conjunto de datos de circuitos.

GrandPrix	AvgSpeed	DistanceDRS	Distance	MaxSpeed	MinSpeed	MaxThrottle	C50	C100	C150	C200	C250	S250	S300	S150	S200	S50
Bahrain Grand Prix	0.542321	0.905769	0.746587	0.474576	0.421875	0.569480	0.2	0.166667	0.500000	0.25	0.0	1.00	0.6	0.000000	0.00	0.0
Emilia Romagna Grand Prix	0.751455	0.414332	0.572921	0.593220	1.000000	0.849160	0.0	0.083333	0.666667	0.75	0.0	0.25	1.0	0.333333	0.25	0.0
Portuguese Grand Prix	0.483543	0.417239	0.477110	0.457627	0.468750	0.492752	0.0	0.333333	0.166667	0.25	0.0	0.75	0.4	0.000000	0.25	0.0
Spanish Grand Prix	0.550184	0.594259	0.490260	0.491525	0.656250	0.560614	0.0	0.166667	0.833333	0.00	0.0	0.50	0.4	0.333333	0.50	0.0
Monaco Grand Prix	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.0	0.333333	0.666667	0.50	0.0	0.75	0.2	1.000000	1.00	1.0

Figura 6.3: Primeros elementos del conjunto de datos de circuitos normalizados aplicando el escalador MinMax.

Para tener una idea de la distribución del conjunto de datos en un plano, se

aplica el algoritmo PCA [67] del paquete Scikit Learn [62]. Este algoritmo permite reducir a un número fijo las dimensiones del conjunto de datos. Por lo tanto, se establece a dos el número de componentes para mostrar el conjunto de datos en un plano. En la figura 6.4 se muestran los dos componentes principales (X e Y) del conjunto de datos de circuitos tras aplicar el algoritmo PCA.

GrandPrix	X	Y
Bahrain Grand Prix	-0.268862	-0.046240
Emilia Romagna Grand Prix	-0.494459	-0.210707
Portuguese Grand Prix	-0.047437	-0.208887
Spanish Grand Prix	0.034468	-0.276424
Monaco Grand Prix	2.158253	-0.681782
Azerbaijan Grand Prix	0.968001	0.924370
French Grand Prix	0.082050	0.502768
Austrian Grand Prix	-0.587742	-0.223465
British Grand Prix	-0.538982	0.040758
Hungarian Grand Prix	0.027047	-0.907248
Dutch Grand Prix	-0.048125	-0.567403
Italian Grand Prix	-0.629788	0.637783
Turkish Grand Prix	0.055985	0.220237
United States Grand Prix	-0.197774	-0.126177
Mexico City Grand Prix	0.951748	0.815042
São Paulo Grand Prix	-0.089592	-0.015319
Qatar Grand Prix	-0.441942	-0.182835
Saudi Arabian Grand Prix	-0.553753	0.408833
Abu Dhabi Grand Prix	-0.379097	-0.103303

Figura 6.4: Elementos del conjunto de datos de circuitos tras la reducción de dimensión aplicando el algoritmo PCA.

El resultado de la reducción de dimensión del conjunto de datos de circuitos se puede observar en la figura 6.5. A partir de esta visualización, se distinguen que la mayoría de circuitos son similares entre ellos.

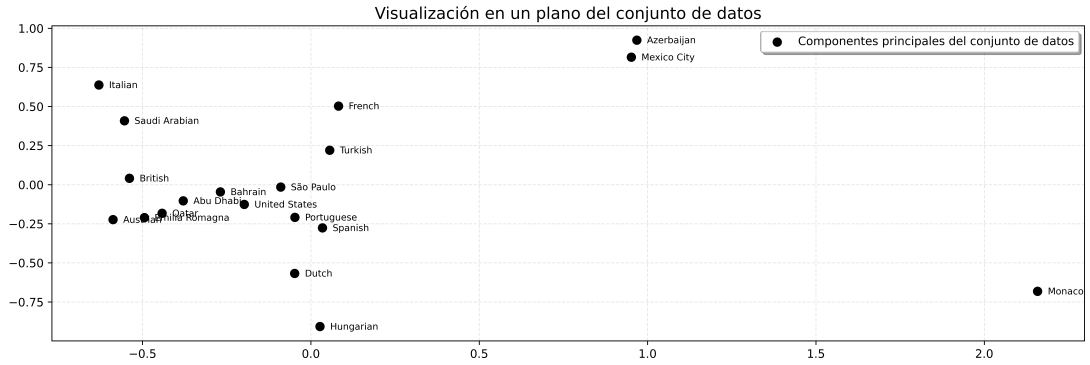

Figura 6.5: Visualización de los elementos del conjunto de datos de circuitos.

6.2.1.1. Optimización de grupos

El algoritmo de *K-Means* necesita un número fijo de grupos en los que agrupar el conjunto de datos. Para encontrar el número de grupos óptimo, se ejecutan dos métodos o técnicas diferentes:

- ***Elbow Method* o Codo de Jambú**: Este método permite establecer el número de grupos óptimo a partir de la inercia del algoritmo de *K-Means*. La inercia es la suma de las distancias al cuadrado de los datos al centroide más cercano. El número de grupos óptimo será aquel en el que la inercia no disminuya de forma drástica.

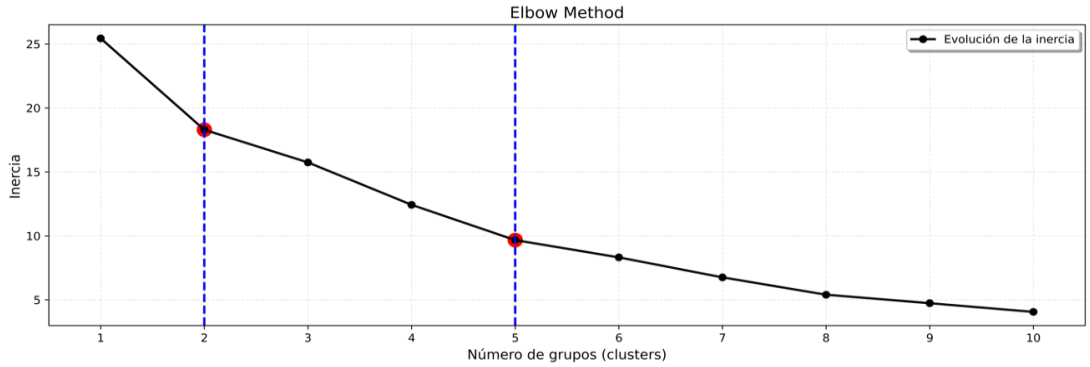

Figura 6.6: Visualización de la evolución de la inercia tras aplicar el método del Codo de Jambú al conjunto de datos.

Como se muestra en la figura 6.6, se puede observar a simple vista que existe un ligero cambio de la inercia con respecto al número de grupos cuando `k = [2, 5]`.

- **Coeficiente de silueta**: Permite medir cómo de similar es un dato con respecto al resto de datos pertenecientes a su grupo, comparado con los datos del grupo más cercano. Para hallar este coeficiente se ha utilizado la métrica del coeficiente de silueta [68] presente en el paquete Scikit Learn [62].

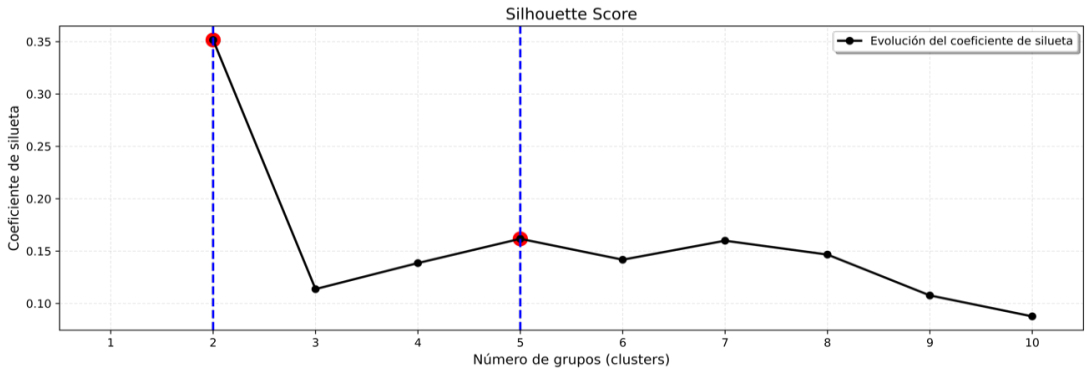

Figura 6.7: Visualización de la evolución del coeficiente de silueta dependiendo del número de grupos.

En la figura 6.7 se puede ver que el coeficiente de silueta cuando `k = 2` es superior al resto. Sin embargo, si se aumenta el número de grupos, los datos más similares entre sí con respecto a otros grupos sucede cuando `k = 5`.

A partir de las métricas recogidas se puede establecer que el número óptimo de grupos es dos, aunque si se desea tener más de una agrupación binaria, la mejor opción es tener cinco grupos diferentes. Es por ello que a continuación, se muestran los resultados de la agrupación del conjunto de datos aplicando el algoritmo de *K-Means* para `k = 2` y `k = 5`.

6.2.1.2. Análisis con 2 *clusters*

En primer lugar, se aplica el algoritmo de *K-Means* para un número fijo de grupos `k = 2`. En la figura 6.8 se muestra los componentes principales del conjunto de datos en un plano, donde se destacan por colores los diferentes grupos obtenidos.

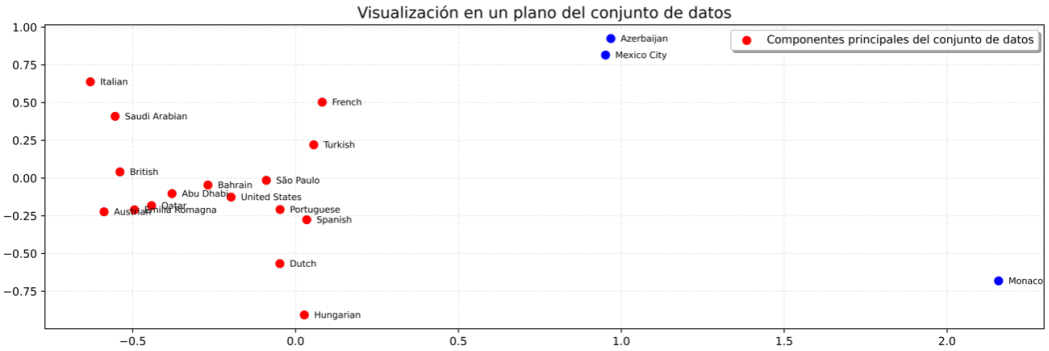

Figura 6.8: Visualización del conjunto de datos con dos *clusters* tras aplicar el algoritmo de *K-Means*.

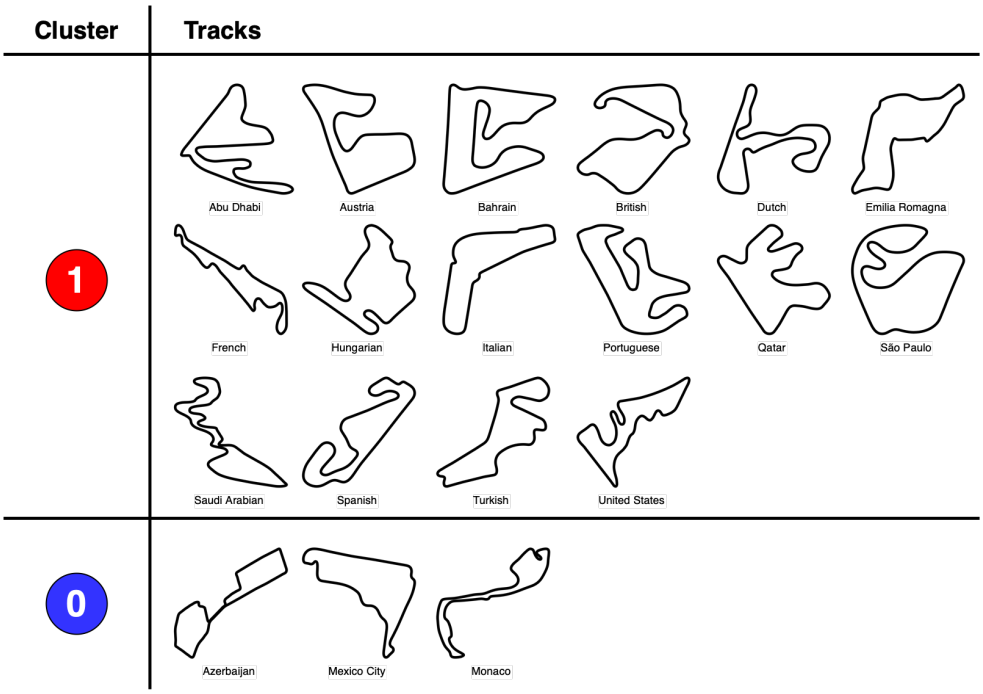

Figura 6.9: Panel de los distintos circuitos agrupados en dos grupos.

El resultado de esta agrupación establece una diferencia entre circuitos con curvas muy lentas (Mónaco, Azerbaiyán y México) con respecto a otros circuitos más rápidos dentro del calendario de la Fórmula 1. En la figura 6.9 se muestra un panel con los circuitos pertenecientes a cada grupo.

6.2.1.3. Análisis con 5 *clusters*

En segundo lugar, se aplica el algoritmo de *K-Means* para un número fijo de grupos `k = 5`. En la figura 6.10 se muestra los componentes principales del conjunto de datos en un plano, donde se destacan por colores los diferentes grupos obtenidos.

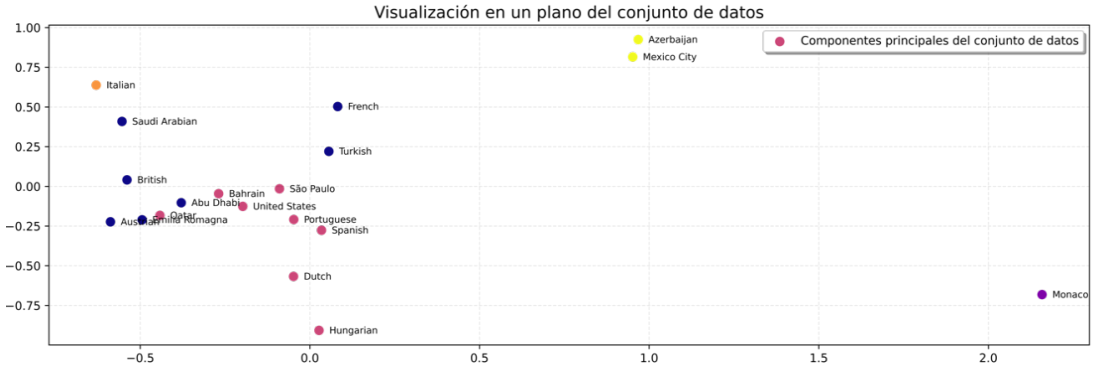

Figura 6.10: Visualización del conjunto de datos con cinco *clusters* tras aplicar el algoritmo de *K-Means*.

En esta ocasión, se distinguen en un único grupo el circuito más rápido (Italia) y el circuito más lento (Mónaco) del calendario oficial de la Fórmula 1. Por otro lado, los circuitos de Azerbaiyán y México se encuentran en el mismo grupo ya que presentan múltiples similitudes: rectas largas y zona de curvas lentas. El resto de grupos divide a los circuitos con más rectas y, por lo tanto, más rápidos (Gran Bretaña o Arabia Saudí) de los circuitos con más curvas y, por lo tanto, más lentos (España o Hungría). A continuación se visualiza en la figura 6.11 un panel con los circuitos pertenecientes a cada grupo.

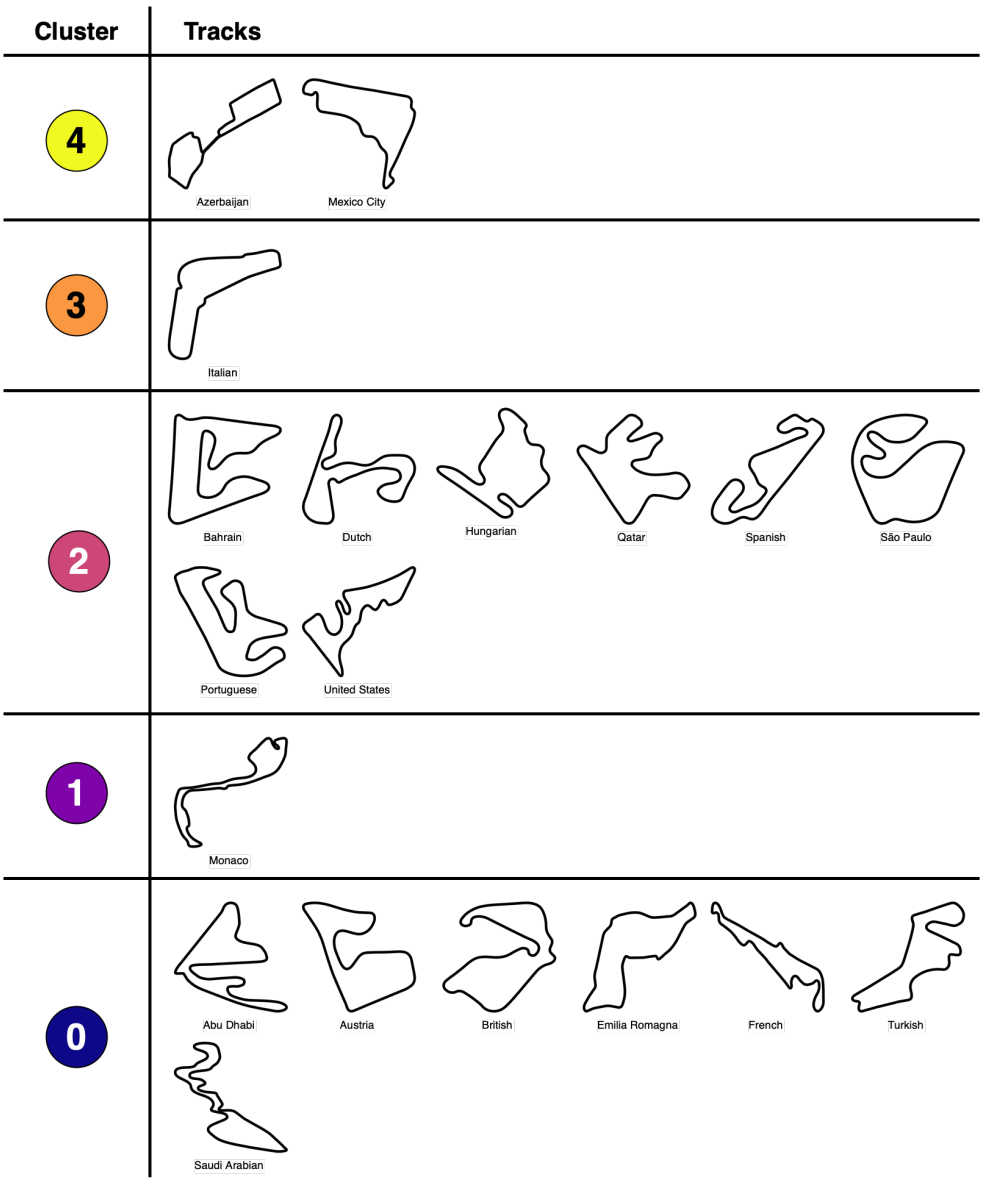

Figura 6.11: Panel de los distintos circuitos agrupados en cinco grupos.

6.2.2. Predicción de resultados

En este apartado se explica con detalle la creación y entrenamiento de los modelos de redes neuronales artificiales de clasificación que permiten predecir los resultados de los pilotos en una carrera de Fórmula 1. Para ello, se utiliza el conjunto de datos obtenido en la sección 6.1.2.

Este conjunto de datos presenta varios datos no numéricos que deben ser transformados. Por lo tanto, se hace uso del archivo auxiliar extraído en la sección 6.1.3 para obtener un identificador único de cada piloto, escudería, circuito y tipo de compuesto del neumático. El resultado de esta transformación se puede observar en la figura 6.12.

	Driver	Team	GrandPrix	Laps	Length	LapTime	Compound	AvgSpeed	MaxSpeed	QualiPos	RacePos
0	0	0	0	56.0	5378.502778	89.385	0	217.504021	314	2.0	1.0
1	1	1	0	56.0	5378.502778	88.997	0	218.342318	315	1.0	2.0
2	2	0	0	56.0	5378.502778	89.586	0	216.873995	312	3.0	3.0
3	3	2	0	56.0	5378.502778	89.974	0	215.943850	316	7.0	4.0
4	4	1	0	56.0	5378.502778	90.659	1	214.904000	310	0.0	5.0

Figura 6.12: Primeros elementos del conjunto de datos de pilotos tras la transformación de los datos no numéricos.

Además, se realiza una copia del conjunto de datos que modifica la columna *RacePos* para categorizarla entre 0 o 1 dependiendo si el piloto ha conseguido puntos durante la carrera. Por lo general, en un Gran Premio de Fórmula 1 los 10 primeros pilotos de la carrera obtienen puntos para el campeonato, mientras que los 10 últimos no consiguen ningún punto. Por otro lado se añade dos nuevas columnas, *Cluster2* y *Cluster5*, que indican el grupo del circuito obtenido tras aplicar el algoritmo de *K-Means* en la sección 6.2.1.

En total se obtienen cuatro conjuntos de datos diferentes: dos *datasets* por puntos y dos *datasets* por posición:

- **Conjuntos de datos por puntos**: Contienen los resultados de los pilotos categorizados dependiendo si han obtenido puntos tras la carrera. El primer conjunto de datos de este tipo presenta la información obtenida en la sección 6.2.1.2 de los circuitos agrupados por dos grupos, y el segundo de ellos contiene la información obtenida en la sección 6.2.1.3 de los circuitos

agrupados por cinco grupos.

- **Conjuntos de datos por posición**: Contienen los resultados individuales de cada piloto tras una carrera. El primer conjunto de datos de este tipo presenta la información obtenida en la sección 6.2.1.2 de los circuitos agrupados por dos grupos, y el segundo de ellos contiene la información obtenida en la sección 6.2.1.3 de los circuitos agrupados por cinco grupos.

A continuación, se normalizan los datos de los conjuntos de datos a través del escalador MinMax [66] del paquete Scikit Learn [62]. En la figura 6.13 se muestra uno de los cuatro conjuntos de datos obtenidos tras la normalización. En este caso, se visualiza el conjunto de datos por puntos donde los circuitos están agrupados en dos grupos.

	Driver	Team	GrandPrix	Laps	Length	LapTime	Compound	AvgSpeed	MaxSpeed	QualiPos	RacePos	Track
0	0.000000	0.000000	0.0	0.214286	0.747557	0.372691	0.0	0.834508	0.889518	0.10	0.0	1.0
1	0.052632	0.111111	0.0	0.214286	0.747557	0.367057	0.0	0.837724	0.892351	0.05	0.0	1.0
2	0.105263	0.000000	0.0	0.214286	0.747557	0.375610	0.0	0.832091	0.883853	0.15	0.0	1.0
3	0.157895	0.222222	0.0	0.214286	0.747557	0.381244	0.0	0.828522	0.895184	0.35	0.0	1.0
4	0.210526	0.111111	0.0	0.214286	0.747557	0.391191	1.0	0.824532	0.878187	0.00	0.0	1.0

Figura 6.13: Primeros elementos del conjunto de datos de pilotos normalizado con circuitos agrupados en dos *clusters*.

6.2.2.1. Estructura del modelo

La estructura de un modelo de red neuronal artificial presenta múltiples hiperparámetros que se pueden modificar para optimizar al máximo los resultados de la clasificación.

En este caso, se utiliza la biblioteca KerasTuner [43] para la optimización y afinación de hiperparámetros y los paquetes TensorFlow y Keras [60, 61] para

construir y entrenar los modelos de aprendizaje profundo. Los hiperparámetros que se optimizan durante la construcción del modelo son los siguientes:

- El número de capas densas de la red neuronal artificial: entre 1 y 5.
- El número de neuronas por cada capa densa de la red neuronal artificial: entre 16 y 256.
- El tipo de función de activación de cada capa de la red neuronal artificial: *ReLU* o *Sigmoid* [69, 70].
- El porcentaje de *dropout* [71] después de cada capa oculta de la red neuronal artificial: entre 0 y 0'3.
- La velocidad de aprendizaje (*learning rate*) [72] del optimizador de la red neuronal artificial: 0'1, 0'01 o 0'001.

```
Search space summary
Default search space size: 5
num_layers (Int)
{'default': None, 'conditions': [], 'min_value': 2, 'max_value': 6, 'step': 1, 'sampling': None}
units_1 (Int)
{'default': None, 'conditions': [], 'min_value': 16, 'max_value': 256, 'step': 16, 'sampling': None}
act_1 (Choice)
{'default': 'relu', 'conditions': [], 'values': ['relu', 'sigmoid'], 'ordered': False}
dropout_1 (Float)
{'default': 0.0, 'conditions': [], 'min_value': 0.0, 'max_value': 0.3, 'step': 0.05, 'sampling': None}
learning_rate (Choice)
{'default': 0.1, 'conditions': [], 'values': [0.1, 0.01, 0.001], 'ordered': True}
```

Figura 6.14: Rango de búsqueda de hiperparámetros para la optimización del modelo de red neuronal artificial.

En la figura 6.14 se muestra el rango de búsqueda de hiperparámetros para la optimización del modelo de red neuronal artificial. Como algoritmo de búsqueda se utiliza el optimizador bayesiano, cuyo objetivo es reducir el número de combinaciones de hiperparámetros a través de un modelo probabilístico [73].

Se establece como función de pérdida (la cual indica a la red neuronal artificial cuando aprender) la métrica de entropía cruzada. Esta métrica mide la diferencia

entre una función de probabilidad sobre la variable de interés y la distribución probabilística real [74].

Por último, se define la capa de salida de la red neuronal artificial. Esta capa tendrá un número de neuronas igual al número de categorías que se quieren clasificar. La función de activación de esta capa se denomina *softmax* [75], donde la suma de todas sus neuronas es igual a 1. Esta función permite obtener el porcentaje de confianza de la elección de una categoría sobre otra.

6.2.2.2. División de los conjuntos de datos

Antes de comenzar con la búsqueda del modelo de red neuronal artificial más óptimo, se debe dividir los conjuntos de datos entre entrenamiento y validación. El conjunto de datos de entrenamiento contiene la información necesaria para que la red neuronal artificial pueda aprender a partir de los datos, mientras que el conjunto de datos de validación permite evaluar la precisión del modelo entrenado.

Para este análisis se ha decidido utilizar el 80 % de los datos para entrenar los modelos de aprendizaje profundo, mientras que el 20 % restante serán destinados a validar el modelo entrenado. Los datos de entrenamiento pertenecen a los dieciséis primeros Grandes Premios y los datos de validación se corresponden con los últimos cuatro de la temporada.

6.2.2.3. Clasificación por puntos

La clasificación por puntos consta de dos conjuntos de datos diferentes. El primero de ellos contiene la información del agrupamiento de circuitos en dos grupos, mientras que el segundo presenta la información del agrupamiento de circuitos en cinco grupos.

Para comenzar, se ejecuta el algoritmo de búsqueda para construir el modelo secuencial de red neuronal artificial con los hiperparámetros más óptimos para el conjunto de datos con dos grupos. En la figura 6.15 se puede observar el resultado de esta búsqueda, donde el modelo más óptimo tiene tres capas densas de 176, 128 y 16 neuronas respectivamente.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten_1 (Flatten)         (None, 11)                0

 dense_2 (Dense)             (None, 176)               2112

 dropout_1 (Dropout)         (None, 176)               0

 dense_3 (Dense)             (None, 128)               22656

 dropout_2 (Dropout)         (None, 128)               0

 dense_4 (Dense)             (None, 16)                2064

 dropout_3 (Dropout)         (None, 16)                0

 dense_5 (Dense)             (None, 2)                 34

=================================================================
Total params: 26,866
Trainable params: 26,866
Non-trainable params: 0
```

Figura 6.15: Estructura del modelo optimizado a partir del *dataset* que contiene el agrupamiento de circuitos en dos grupos.

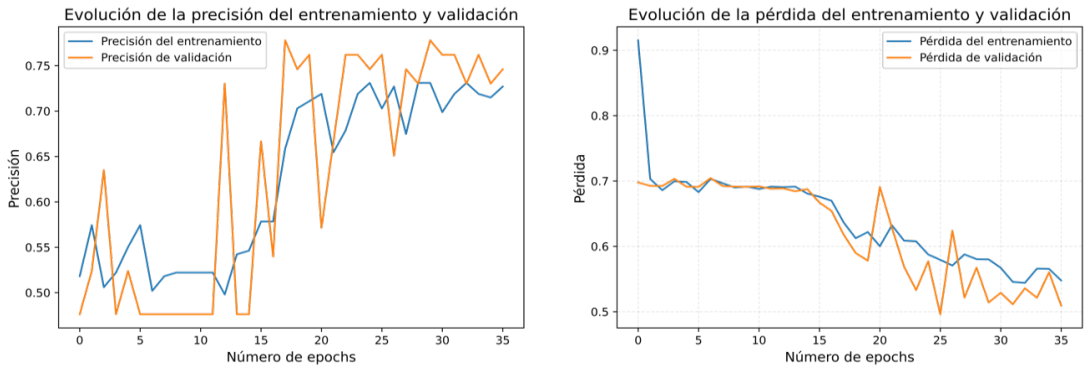

Figura 6.16: Evolución de la precisión y función de pérdida de este modelo durante el entrenamiento.

A continuación, se evalúa el modelo más óptimo entrenado con los datos de validación. Como se puede visualizar en la figura 6.16, la precisión de este modelo de red neuronal es del 74 % con los datos de validación.

Por otro lado, se ejecuta el algoritmo de búsqueda para construir el modelo con el conjunto de hiperparámetros más óptimos para el conjunto de datos con cinco grupos. El resultado de esta búsqueda se observa en la figura 6.17, donde se aprecia que el modelo más óptimo está compuesto únicamente por una capa densa de 256 neuronas.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten_1 (Flatten)         (None, 11)                0

 dense_2 (Dense)             (None, 256)               3072

 dropout_1 (Dropout)         (None, 256)               0

 dense_3 (Dense)             (None, 2)                 514

=================================================================
Total params: 3,586
Trainable params: 3,586
Non-trainable params: 0
```

Figura 6.17: Estructura del modelo optimizado a partir del *dataset* que contiene el agrupamiento de circuitos en cinco grupos.

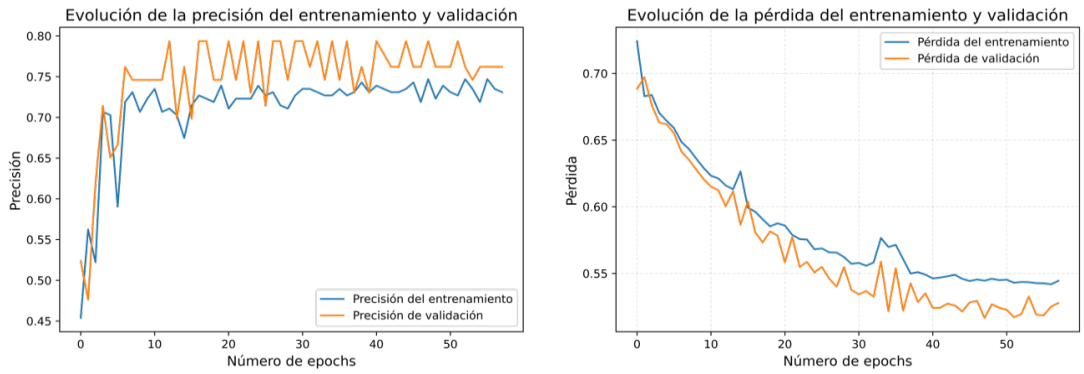

Figura 6.18: Evolución de la precisión y función de pérdida de este modelo durante el entrenamiento.

En la figura 6.18 se muestra la evolución de este modelo de red neuronal artificial durante el proceso de entrenamiento. El resultado obtenido al evaluar los datos de validación es del 73 % de precisión.

Los resultados obtenidos tras evaluar los modelos de redes neuronales artificiales, para clasificar los resultados de un piloto a partir de los puntos conseguidos en una carrera, son del 74 % con los circuitos agrupados en dos *clusters* y del 73 % con los circuitos agrupados en cinco *clusters*.

	loss	accuracy
Model with 2 clusters	0.51197	0.746835
Model with 5 clusters	0.51893	0.734177

Figura 6.19: Comparación de la precisión de la predicción por puntos de los modelos entrenados.

Este resultado no es muy bueno si se quiere predecir con exactitud el resultado de un piloto, pero debido a la amplia variabilidad del mundo de la Fórmula 1, permite ser un elemento de ayuda a la hora de realizar comentarios, encuestas o paneles de información sobre una carrera.

6.2.2.4. Clasificación por posición

La clasificación por posición consta de dos conjuntos de datos diferentes. El primero de ellos contiene la información del agrupamiento de circuitos en dos grupos, mientras que el segundo presenta la información del agrupamiento de circuitos en cinco grupos.

Para comenzar, se ejecuta el algoritmo de búsqueda para construir el modelo secuencial de red neuronal artificial con los hiperparámetros más óptimos para el

conjunto de datos con dos grupos. En la figura 6.20 se puede observar el resultado de esta búsqueda, donde el modelo más óptimo tiene tres capas densas de 96, 16 y 16 neuronas respectivamente.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten_1 (Flatten)         (None, 11)                0

 dense_6 (Dense)             (None, 96)                1152

 dropout_5 (Dropout)         (None, 96)                0

 dense_7 (Dense)             (None, 16)                1552

 dropout_6 (Dropout)         (None, 16)                0

 dense_8 (Dense)             (None, 16)                272

 dropout_7 (Dropout)         (None, 16)                0

 dense_9 (Dense)             (None, 20)                340

=================================================================
Total params: 3,316
Trainable params: 3,316
Non-trainable params: 0
```

Figura 6.20: Estructura del modelo optimizado a partir del *dataset* que contiene el agrupamiento de circuitos en dos grupos.

A continuación, se evalúa el modelo más óptimo entrenado con los datos de validación. Como se puede visualizar en la figura 6.21, la precisión de este modelo de red neuronal artificial es del 11 % con los datos de validación.

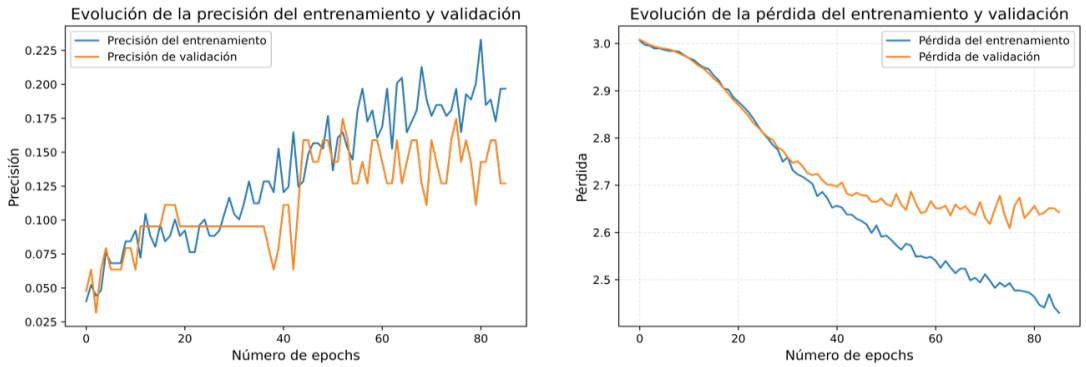

Figura 6.21: Evolución de la precisión y función de pérdida de este modelo durante el entrenamiento.

Por otro lado, se ejecuta el algoritmo de búsqueda para construir el modelo con el conjunto de hiperparámetros más óptimos para el conjunto de datos con cinco grupos. El resultado de esta búsqueda se observa en la figura 6.22, donde se aprecia que el modelo más óptimo está compuesto por tres capas densas de 256, 16 y 256 neuronas respectivamente.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 flatten_1 (Flatten)          (None, 11)                0

 dense_4 (Dense)              (None, 256)               3072

 dropout_3 (Dropout)          (None, 256)               0

 dense_5 (Dense)              (None, 16)                4112

 dropout_4 (Dropout)          (None, 16)                0

 dense_6 (Dense)              (None, 256)               4352

 dropout_5 (Dropout)          (None, 256)               0

 dense_7 (Dense)              (None, 20)                5140

=================================================================
Total params: 16,676
Trainable params: 16,676
Non-trainable params: 0
```

Figura 6.22: Estructura del modelo optimizado a partir del *dataset* que contiene el agrupamiento de circuitos en cinco grupos.

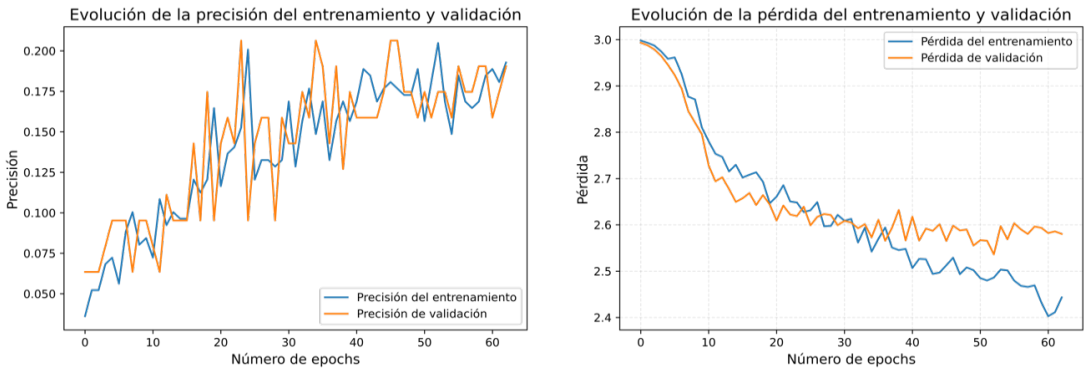

Figura 6.23: Evolución de la precisión y función de pérdida de este modelo durante el entrenamiento.

En la figura 6.23 se muestra la evolución del modelo de red neuronal artificial durante el proceso de entrenamiento. El resultado obtenido al evaluar los datos de validación es del 8 % de precisión.

Los resultados obtenidos tras evaluar los modelos de redes neuronales artificiales, para clasificar los resultados de un piloto a partir de la posición obtenida en una carrera, son del 11 % con los circuitos agrupados en dos *clusters* y del 8 % con los circuitos agrupados en cinco *clusters*.

	loss	accuracy
Model with 2 clusters	2.764458	0.113924
Model with 5 clusters	2.735952	0.088608

Figura 6.24: Comparación de la precisión de la predicción por posición de los modelos entrenados.

Debido a la gran variabilidad existente en la Fórmula 1 (problemas de fiabilidad, meteorología cambiante, accidentes...) es muy difícil calcular la posición exacta que obtendrá un piloto. A comparación de predecir una posición aleatoria (5 %) en una carrera, este modelo consigue únicamente el doble como mejor resultado. Esto es un claro ejemplo del constante cambio alrededor de este deporte, y lo difícil que es predecir el resultado individual de un piloto en una carrera.

6.3. Aplicación de visualización

En este apartado se define la estructura de la aplicación web planteada. Esta aplicación tiene como finalidad visualizar los datos de telemetría de Fórmula 1 disponibles, para que cualquier aficionado de este deporte pueda acceder a ellos de forma dinámica e interactiva.

6.3.1. Requisitos

A continuación se indican los requisitos funcionales y no funcionales definidos para desarrollar la aplicación web.

6.3.1.1. Requisitos funcionales

RF-01	**Creación de una API**
Versión	1.0
Autor	David Morán Montero
Descripción	Se crea una API para obtener y transformar los datos de Fórmula 1 que se necesitan para su visualización en la aplicación.
Comentarios	La API se crea usando el *framework* FastAPI [25] para obtener los datos a través del paquete FastF1 [57].

Tabla 6.1: RF-01: Creación de una API.

RF-02	**Visualización de los datos de telemetría**
Versión	1.0
Autor	David Morán Montero
Descripción	Se visualizan los siguientes datos de telemetría de todos los pilotos desde la temporada 2018 hasta la actualidad: • Velocidad. • Porcentaje de aceleración. • Número de marchas. • Revoluciones por minuto del motor.
Comentarios	Los datos están en el orden de la distancia del circuito.

Tabla 6.2: RF-02: Visualización de los datos de telemetría.

RF-03	**Análisis general de un Gran Premio**
Versión	1.0
Autor	David Morán Montero
Descripción	Se realiza un análisis de las estadísticas y los datos obtenidos durante un Gran Premio. Los datos que se muestran son los siguientes: • Tiempo de clasificación de los pilotos. • Estrategias durante la carrera: paradas, longitud del *stint*, tipo de compuesto del neumático... • Gráfica de los puntos conseguidos por cada piloto hasta ese Gran Premio.
Comentarios	Ninguno.

Tabla 6.3: RF-03: Análisis general de un Gran Premio.

RF-04	**Predicción de los resultados de una carrera**
Versión	1.0
Autor	David Morán Montero
Descripción	Se muestra el porcentaje de precisión al predecir los resultados de cada piloto durante las últimas cuatro carreras de la temporada 2021.
Comentarios	Un diagrama de barras horizontales muestra el resultado de la predicción si los pilotos consiguen puntos y otro diagrama del mismo tipo muestra el resultado de la evaluación si se predice que el piloto no ha conseguido puntos. Se utilizan los modelos analizados y evaluados durante la sección 6.2.2.3 para la predicción de los resultados de los pilotos.

Tabla 6.4: RF-04: Predicción de los resultados de una carrera.

RF-05	**Actualización de los datos**
Versión	1.0
Autor	David Morán Montero
Descripción	Los datos se actualizan por cada Gran Premio disputado.
Comentarios	Gracias al paquete FastF1 [57], los datos se mantienen actualizados en todo momento.

Tabla 6.5: RF-05: Actualización de los datos.

RF-06	**Reutilización de componentes**
Versión	1.0
Autor	David Morán Montero
Descripción	La estructura interna de la aplicación web es completamente reutilizable.
Comentarios	Debido al uso de React [24] como *framework* para realizar aplicaciones web, se aprovechan todas sus ventajas a la hora de reutilizar los componentes.

Tabla 6.6: RF-06: Reutilización de componentes.

6.3.1.2. Requisitos no funcionales

RNF-01	**Aplicación web dinámica e interactiva**
Versión	1.0
Autor	David Morán Montero
Descripción	La aplicación web es dinámica (se puede modificar la visualización de los datos en todo momento) e interactiva (los datos son interactivos y añaden funcionalidades extra).
Comentarios	Ninguno.

Tabla 6.7: RNF-01: Aplicación web dinámica.

RNF-02	**Disponibilidad de los datos**
Versión	1.0
Autor	David Morán Montero
Descripción	Se guardan los datos en una caché para mantener la disponibilidad del servicio cuando haya problemas con el paquete FastF1 [57].
Comentarios	Ninguno.

Tabla 6.8: RNF-02: Disponibilidad de los datos.

RNF-03	**Diseño uniforme**
Versión	1.0
Autor	David Morán Montero
Descripción	El diseño de la aplicación web es uniforme en todas sus pestañas.
Comentarios	De esta manera, el usuario no tendrá problemas de uso en ninguna de las páginas de la aplicación.

Tabla 6.9: RNF-03: Diseño uniforme.

6.3.2. Procesos de negocio

El modelado del negocio permite comprender la estructura y la dinámica existente en un sistema software. Su finalidad es describir cada proceso del negocio, especificando sus datos, actividades, roles y reglas de negocio. Para realizar el modelado del negocio se utiliza la notación gráfica *Business Process Model and Notation* (BPMN) [76].

6.3.2.1. Visualización de datos de telemetría por piloto

El usuario selecciona los datos de búsqueda correspondientes: temporada, circuito, sesión y piloto. A continuación, se realiza una llamada a la API que se encarga de obtener los datos a partir de la caché (si están guardados de antemano) y devolverlos para realizar la visualización de los tiempos por vuelta del piloto durante la sesión. Si el usuario decide elegir una vuelta concreta, se realiza una segunda petición a la API que devuelve los datos de telemetría de la vuelta seleccionada para su posterior visualización.

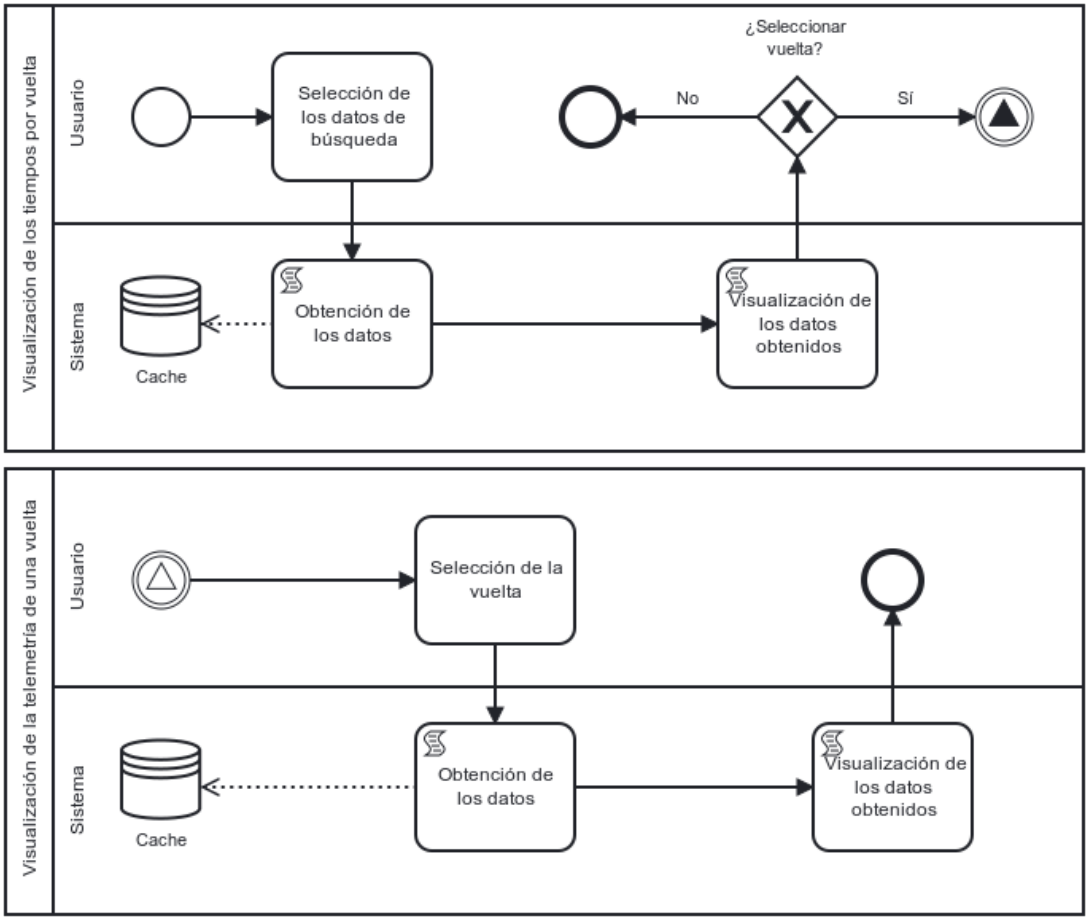

Figura 6.25: Modelo de procesos de negocio de la visualización de datos de telemetría por piloto.

6.3.2.2. Generación de estadísticas generales de un Gran Premio

Para la generación de estadísticas generales de un Gran Premio, el usuario debe seleccionar la temporada y el circuito a analizar. A continuación, el sistema realiza una petición a la API para obtener los datos a partir de la caché (si están guardados de antemano). Además, se realiza un procesamiento de los datos para generar las estadísticas correspondientes antes de obtener la visualización.

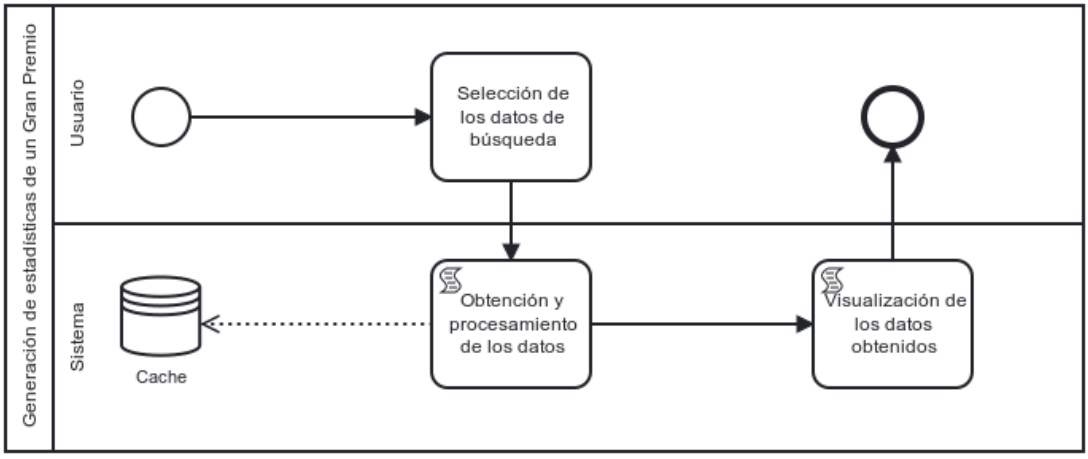

Figura 6.26: Modelo de procesos de negocio de la generación de estadísticas de un Gran Premio.

6.3.2.3. Predicción de los resultados de una carrera

El usuario selecciona los datos de búsqueda para realizar la predicción: uno de los cuatro últimos Grandes Premios de la temporada 2021. A continuación, el sistema selecciona el modelo y el conjunto de datos correspondiente para realizar la evaluación de los datos. Finalmente, el sistema devuelve la visualización de la precisión de la predicción de los resultados de los pilotos durante una carrera.

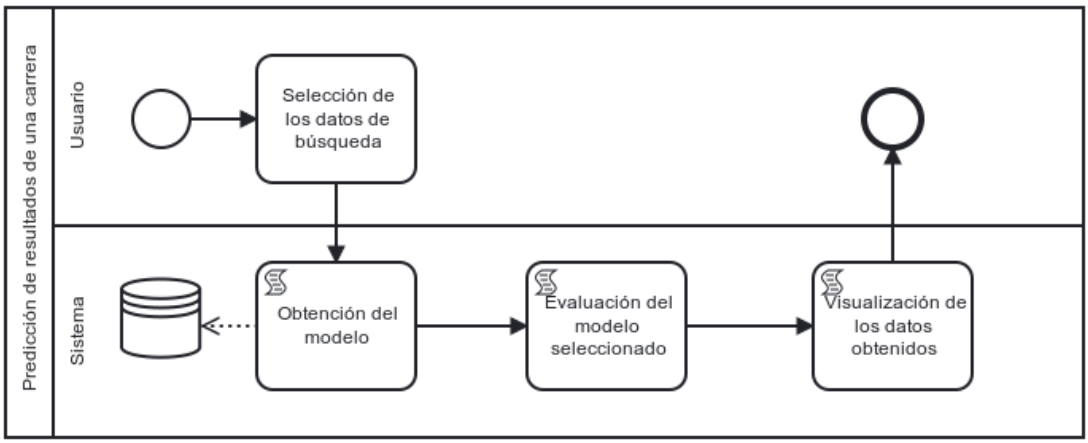

Figura 6.27: Modelo de procesos de negocio de la visualización de datos de telemetría por piloto.

6.3.3. Casos de uso

En esta sección se definen los diferentes casos de uso para describir las actividades de la aplicación web. En la figura 6.28 se muestra el diagrama de casos de uso de la aplicación web.

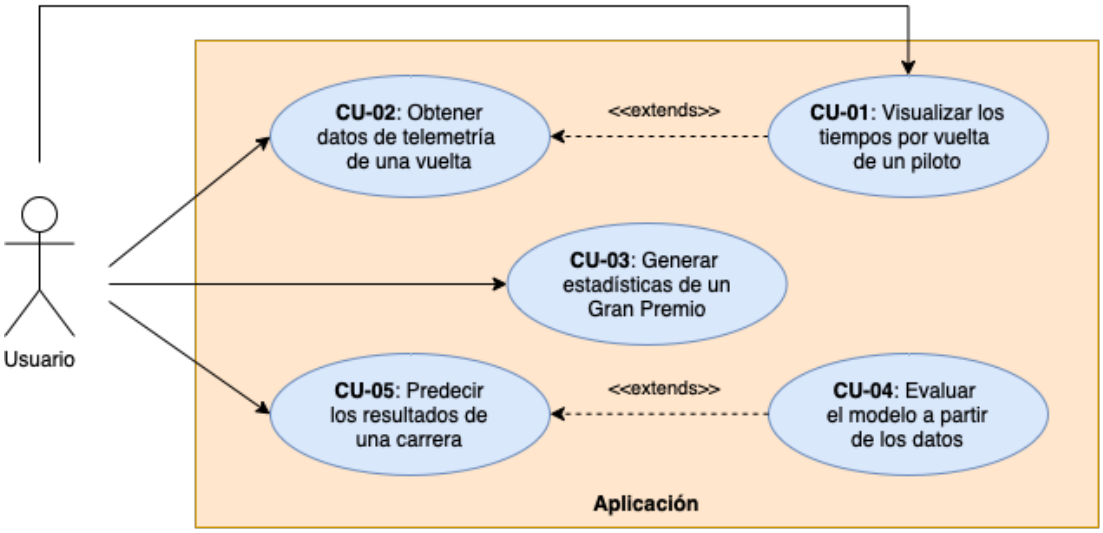

Figura 6.28: Diagrama de casos de uso de la aplicación web.

A continuación se definen los diferentes casos de uso de la aplicación web que se muestran en el diagrama de casos de uso:

CU-01	**Visualizar los tiempos por vuelta de un piloto**	
Versión	1.0	
Autor	David Morán Montero	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario quiere visualizar los tiempos por vuelta de un piloto durante una sesión de un Gran Premio de Fórmula 1.	
Precondición	La API debe estar en funcionamiento.	
Secuencia normal	Paso	Acción
	1	El usuario realiza la selección de los datos de búsqueda: temporada, circuito, sesión y piloto.
	2	Se realiza la petición a la API para obtener los tiempos por vuelta del piloto seleccionado.
	3	Los datos obtenidos se visualizan en la aplicación web.
Postcondición	Los tiempos por vuelta del piloto seleccionado se muestran en la aplicación web.	
Comentarios	Ninguno.	

Tabla 6.10: CU-01: Visualizar los tiempos por vuelta de un piloto.

CU-02	**Obtener datos de telemetría de una vuelta**
Versión	1.0
Autor	David Morán Montero
Continúa en la siguiente página	

CU-02	**Obtener datos de telemetría de una vuelta**	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario quiere visualizar los datos de telemetría de un piloto durante una vuelta.	
Precondición	La API debe estar en funcionamiento y se debe haber realizado el caso de uso CU-01.	
Secuencia normal	Paso	Acción
	1	El usuario realiza la selección de los datos de telemetría: velocidad, porcentaje de aceleración, número de marchas del vehículo o revoluciones por minuto del motor.
	2	Se realiza la petición a la API para obtener los datos de telemetría seleccionados.
	3	Los datos obtenidos se visualizan en la aplicación web.
Postcondición	Los datos de telemetría seleccionados de una vuelta de un piloto se muestran en la aplicación web.	
Comentarios	Ninguno.	

Tabla 6.11: CU-02: Obtener datos de telemetría de una vuelta.

CU-03	**Generar estadísticas de un Gran Premio**
Versión	1.0
Autor	David Morán Montero
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario quiere generar las estadísticas generales de un Gran Premio de una temporada de Fórmula 1.

Continúa en la siguiente página

CU-03	**Generar estadísticas de un Gran Premio**	
Precondición	La API debe estar en funcionamiento.	
Secuencia normal	Paso	Acción
	1	El usuario selecciona los datos de búsqueda: temporada y circuito.
	2	Se realiza la petición a la API que devuelve todos los datos necesarios para generar las estadísticas.
	3	Se visualiza los tiempos de clasificación del Gran Premio a partir de los datos obtenidos.
	4	Se muestra la estrategia de carrera a partir de los datos obtenidos.
	5	Se visualiza los puntos y la clasificación del campeonato de Fórmula 1 hasta la fecha a partir de los datos obtenidos.
Postcondición	Se obtienen estadísticas generales de un Gran Premio de Fórmula 1.	
Comentarios	Ninguno.	

Tabla 6.12: CU-03: Generar estadísticas de un Gran Premio.

CU-04	**Evaluar el modelo a partir de los datos**	
Versión	1.0	
Autor	David Morán Montero	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario quiere predecir los resultados de un piloto en una carrera de Fórmula 1.	
Precondición	La API debe estar en funcionamiento.	
Secuencia	Paso	Acción
Continúa en la siguiente página		

CU-04	**Evaluar el modelo a partir de los datos**	
normal	1	El usuario selecciona los datos de búsqueda: circuito y modelo a evaluar.
	2	Se realiza una petición a la API que recoge y evalúa el modelo seleccionado.
Postcondición	El modelo seleccionado está evaluado a partir de los resultados de los pilotos en el circuito seleccionado.	
Comentarios	Ninguno.	

Tabla 6.13: CU-04: Evaluar el modelo a partir de los datos.

CU-05	**Predecir los resultados de una carrera**	
Versión	1.0	
Autor	David Morán Montero	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario quiere generar predicciones de los resultados de un piloto durante una carrera de Fórmula 1.	
Precondición	La API debe estar en funcionamiento. Se debe de haber evaluado el modelo a partir de los datos seleccionados.	
Secuencia normal	Paso	Acción
	1	Se debe predecir los resultados de cada piloto en una carrera a partir del modelo entrenado.
	2	Se visualizan las distintas predicciones de los resultados de los pilotos en una carrera.
Postcondición	Se muestra el porcentaje de precisión de la predicción.	
Comentarios	Ninguno.	

Tabla 6.14: CU-05: Predecir los resultados de una carrera.

6.3.4. Navegación

La navegación de la aplicación web es bastante sencilla al ser de poco tamaño. La aplicación consta de tres pestañas o páginas que están enlazadas unas a otras en todo momento a través de la barra de navegación. En la figura 6.29 se muestra el diagrama de navegación de la aplicación web desarrollada.

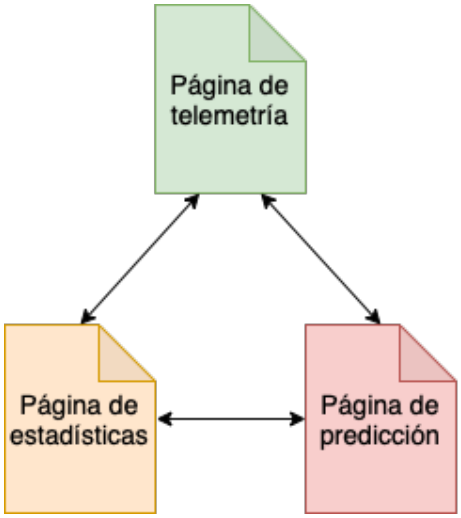

Figura 6.29: Diagrama de navegación de la aplicación web.

La aplicación web esta diseñada para adaptarse a teléfonos móviles, *tablets* y monitores de ordenador. En la figura 6.30 se muestra la barra de navegación adaptada a dispositivos móviles.

Figura 6.30: Barra de navegación de la aplicación web adaptada a dispositivos móviles.

6.3.4.1. Página de telemetría

En la página de telemetría se presenta cuatro elementos de selección para indicar los datos de búsqueda: temporada, circuito, sesión y piloto de Fórmula 1. Como se puede observar en la figura 6.31, los elementos de selección cargan los datos dinámicamente.

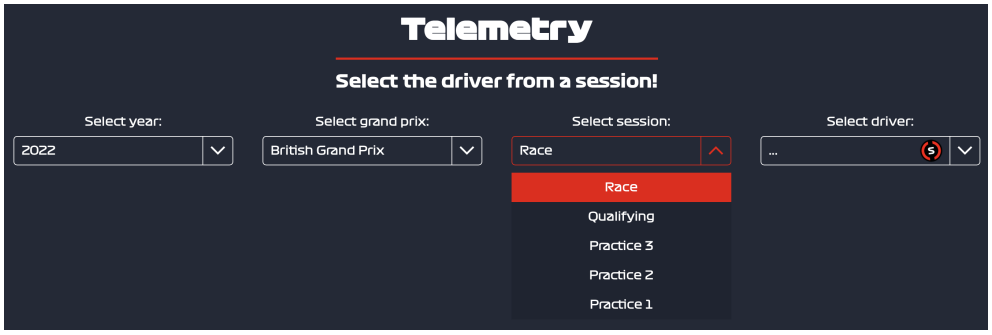

Figura 6.31: Elementos de selección de los datos de búsqueda.

Una vez cargados los datos de búsqueda, se muestra una gráfica de lineas que presenta los tiempos de vuelta de un piloto durante una sesión de un Gran Premio, como se observa en la figura 6.32.

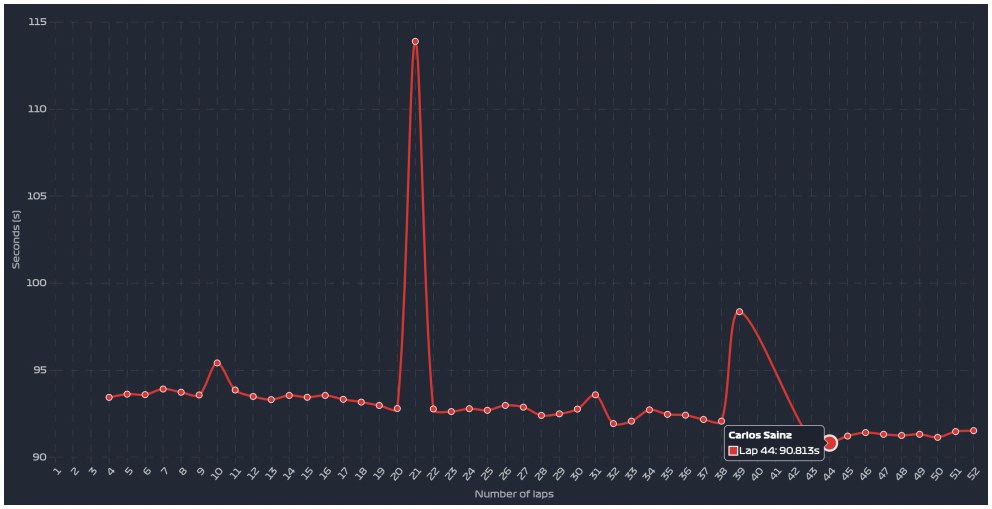

Figura 6.32: Gráfico de líneas de los tiempos por vuelta de un piloto durante una sesión.

Esta gráfica es interactiva, y si se pulsa sobre un punto de la misma, se obtendrá la información de telemetría de la vuelta seleccionada. Existen distintos tipos de datos de telemetría que se pueden obtener: velocidad, porcentaje de aceleración, número de marchas del vehículo y revoluciones por minuto del motor. En la figura 6.33 se observa la velocidad de un piloto durante una vuelta concreta.

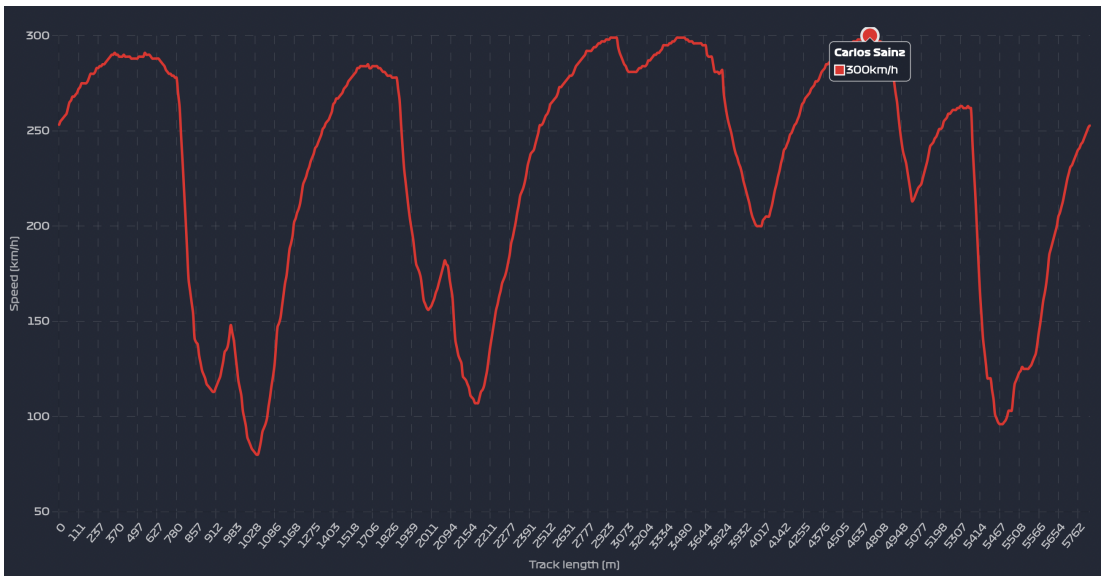

Figura 6.33: Gráfico de líneas de la velocidad de un piloto durante una vuelta concreta.

6.3.4.2. Página de estadísticas

La página de estadísticas es bastante similar a la página de telemetría. En este caso, se debe seleccionar la temporada y el Gran Premio para obtener las estadísticas. En las figuras 6.34, 6.35 y 6.36 se muestran las estadísticas de los tiempos de clasificación, las estrategias de carreras y los puntos y clasificación de pilotos del campeonato durante el Gran Premio seleccionado.

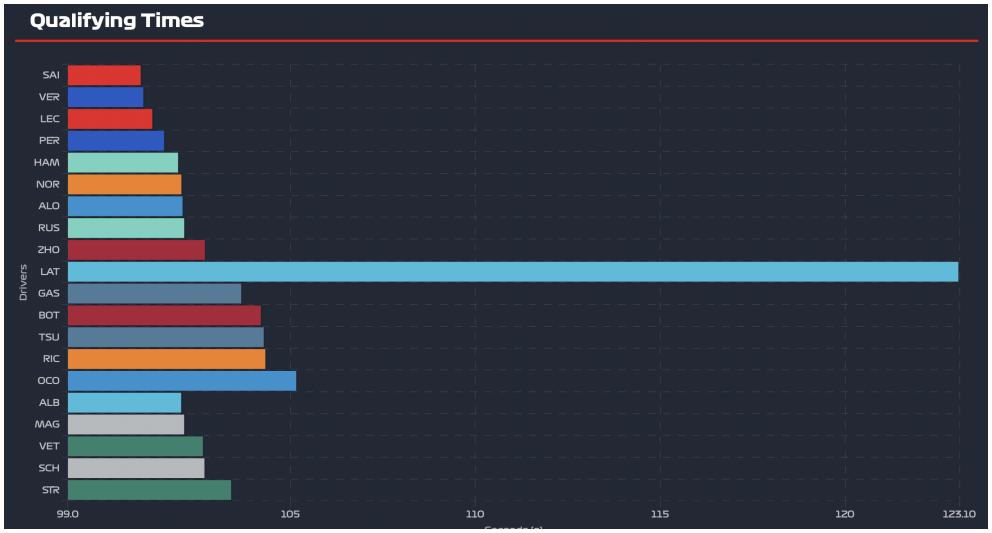

Figura 6.34: Diagrama de barras horizontales con los tiempos de clasificación de los pilotos en un Gran Premio.

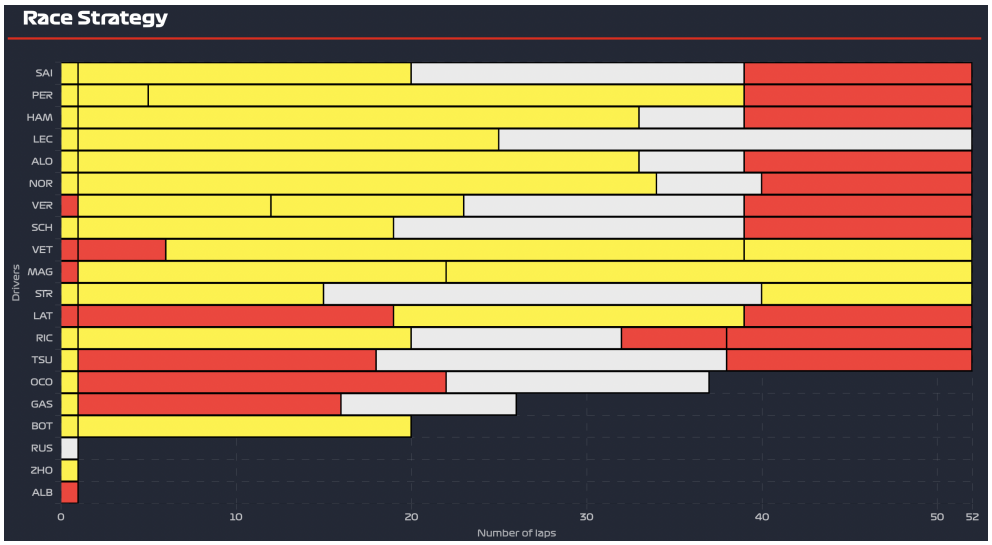

Figura 6.35: Estrategias de carreras de los pilotos durante un Gran Premio.

La clasificación del campeonato viene marcada por la cantidad de puntos que posee un piloto tras finalizar las carreras de Fórmula 1. La siguiente figura muestra los puntos de todos los pilotos desde el comienzo de la temporada hasta el Gran Premio seleccionado para la generación de estadísticas.

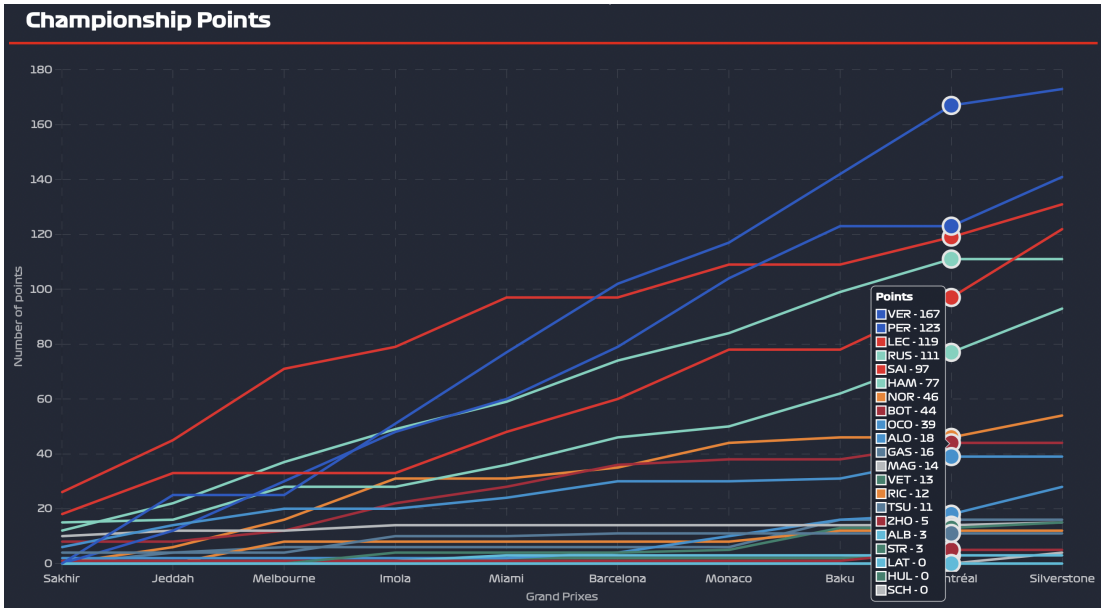

Figura 6.36: Gráfico de líneas con los puntos y clasificación de los pilotos en el campeonato de Fórmula 1.

6.3.4.3. Página de predicción

Por último, la página de predicción muestra el resultado del análisis y evaluación de los modelos de redes neuronales artificiales construidos y entrenados en la sección 6.2.2.3.

En esta página, únicamente se podrá seleccionar uno de los cuatro últimos circuitos del calendario oficial de la Fórmula 1 durante la temporada 2021. En la figura 6.37 se muestra el resultado de la predicción, donde se visualiza la categoría en la que ha clasificado el resultado y el porcentaje de confianza de pertenecer a dicha categoría. Además, se permite ver los resultados del modelo con dos o cinco *clusters* y distinguir si la predicción ha acertado o no.

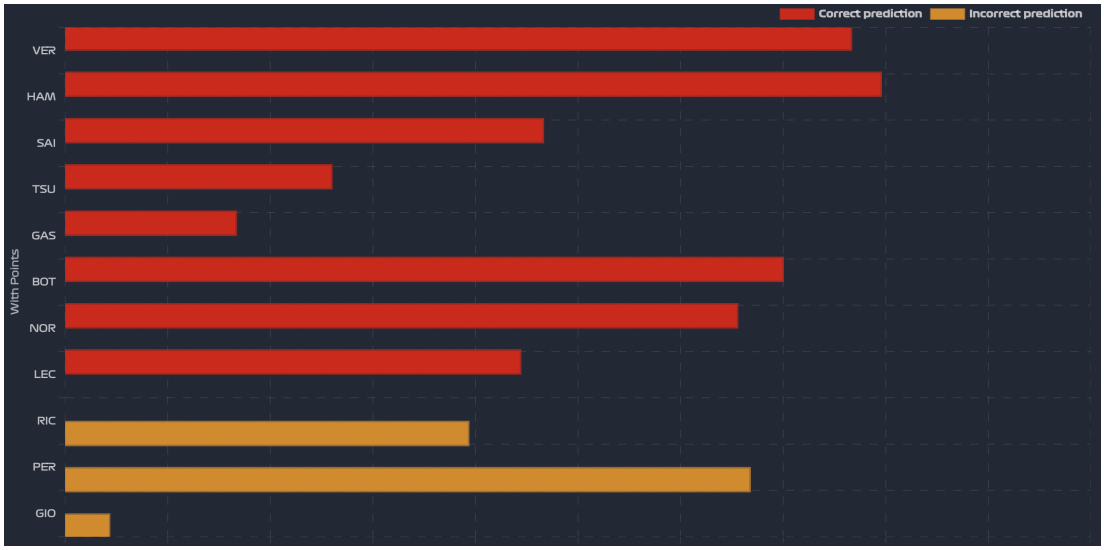

Figura 6.37: Diagrama de barras horizontales que muestran el resultado de la predicción de los modelos entrenados en la sección 6.2.2.3.

7. Conclusiones

La Fórmula 1 es un campo con mucho potencial dentro de la ingeniería de datos. Desde las características de un circuito concreto hasta el estado del vehículo durante una carrera, los datos son necesarios para que las escuderías y los pilotos permitan obtener el mejor rendimiento posible.

En este trabajo se han aplicado algoritmos y creado modelos de aprendizaje automático para la agrupación de los circuitos de Fórmula 1 y la clasificación de los resultados de los pilotos durante una carrera. Además, se ha creado una aplicación web que permite la visualización de datos de telemetría y estadísticas de la Fórmula 1. Gracias a la constante investigación en el ámbito de este trabajo, he conseguido aprender y aplicar mis conocimientos obtenidos durante el grado para analizar, evaluar y mejorar el desarrollo del proyecto.

Durante la realización de este trabajo, se han encontrado varias dificultades que se han ido superando poco a poco: obtención de los datos de telemetría, visualización de los resultados en diferentes plataformas, creación de modelos de aprendizaje profundo con buenos resultados, etc. Aún así, gracias a la amplia investigación realizada se han conseguido solucionar todos los problemas encontrados.

Si se tiene en cuenta la gran variabilidad de este deporte, los resultados obtenidos durante el análisis y evaluación del proyecto son buenos y permiten ser un elemento de ayuda en cualquier encuesta, análisis o panel de información sobre la carrera de un Gran Premio de Fórmula 1. Hay que tener en cuenta que para la

creación y entrenamiento de los modelos de aprendizaje automático se ha utilizado un conjunto de datos bastante pequeño, lo que ha limitado en cierta forma los resultados obtenidos.

La inteligencia artificial en la Fórmula 1 es un campo de estudio que tiene mucho potencial para sacar el mejor rendimiento de los pilotos y vehículos, además de mejorar la experiencia de los espectadores durante un Gran Premio. El desarrollo de este proyecto me ha permitido conocer dos mundos que me apasionan (la ingeniería de datos y la Fórmula 1) y ha conseguido que pueda seguir adelante con mis propios proyectos en un futuro.

7.1. Trabajos futuros

Aunque se han cumplido los objetivos de este proyecto, existen trabajos futuros que se pueden realizar para ampliar y mejorar lo desarrollado durante el trabajo:

- Búsqueda de nuevas fuentes de datos que ayuden a mejorar y ampliar considerablemente los distintos conjuntos de datos.
- Ampliación de las funcionalidades de la aplicación web: nuevas visualizaciones, solución de *bugs*, mejoras en el diseño, puesta en producción, etc.
- Análisis exhaustivo de la predicción de resultados: búsqueda de nuevos algoritmos, mejora de hiperparámetros, ampliación del conjunto de datos, etc.
- Desarrollo de nuevos proyectos dentro de este ámbito que permitan ayudar a las personas interesadas en el mundo de la Fórmula 1.

Bibliografía

[1] Wikipedia, "Archivo: F1.svg," Noviembre 2017. [En línea]. Disponible en: https://es.m.wikipedia.org/wiki/Archivo:F1.svg (accedido en Jun. 27, 2022).

[2] Jasper, "How to Analyze Formula 1 Telemetry in 2022," Marzo 2022. [En línea]. Disponible en: https://miro.medium.com/max/1400/1*8n2y4g7Eqgy9TWv2AQWgRQ.png (accedido en Jun. 27, 2022).

[3] R. Lima, "AI Research Published by Year and Source," Junio 2020. [En línea]. Disponible en: https://www.researchgate.net/profile/Rui-Lima-16/publication/342972286/figure/fig3/AS:917456684085248@1595750127376/AI-Research-Published-by-Year-and-Source.ppm (accedido en Jun. 29, 2022).

[4] D. Gutierrez, "The Difference between AI, Machine Learning and Deep Learning," Febrero 2017. [En línea]. Disponible en: https://insidebigdata.com/wp-content/uploads/2017/02/Deep_Learning_Icons_R5_PNG.jpg.png (accedido en Jun. 29, 2022).

[5] H. Store, "Types of Machine Learning Algorithms." [En línea]. Disponible en: https://miro.medium.com/max/1200/1*FUZS9K4JPqzfXDcC83BQTw.png (accedido en Jun. 29, 2022).

[6] A. Akbari, "Schematic of a Decision Tree," Enero 2021. [En línea]. Disponible en: https://www.researchgate.net/profile/Amir-Akbari-19/publication/348456545 (accedido en Jun. 29, 2022).

[7] Wikipedia, "Archivo:Linear regression.svg," Noviembre 2010. [En línea]. Disponible en: https://es.m.wikipedia.org/wiki/Archivo:Linear_regression.svg (accedido en Jun. 30, 2022).

[8] Wikipedia, "File:GaussianScatterPCA.svg," Febrero 2016. [En línea]. Disponible en: https://en.wikipedia.org/wiki/File:GaussianScatterPCA.svg (accedido en Jun. 30, 2022).

[9] T. Yang, "Clustering," Octubre 2020. [En línea]. Disponible en: https://tyami.github.io/assets/images/post/ML/2020-10-30-clustering/2020-10-30-clustering-11-hard-soft-clustering.png (accedido en Jun. 30, 2022).

[10] B. Yerra, "Centroid-based Clustering K-Means Algorithm," Febrero 2018. [En línea]. Disponible en: https://mlbhanuyerra.github.io/img/Clustering_K-means_files/Clustering_K-means_7_1.png (accedido en Jun. 30, 2022).

[11] H. Store, "Reinforcement learning." [En línea]. Disponible en: https://www.novatec-gmbh.de/wp-content/uploads/1_mPGk9WTNNvp3i4-9JFgD3w.png (accedido en Jun. 30, 2022).

[12] S. Butt, "Artificial Neuron," Agosto 2017. [En línea]. Disponible en: https://rpubs.com/Spencer_Butt_RPubs/301131 (accedido en Jun. 30, 2022).

[13] M. N. Alam, "A typical artificial neural network (ANN)," Agosto 2016. [En línea]. Disponible en: https://www.researchgate.net/profile/Mahamad-Alam/publication/305325563/figure/fig1/AS:384012620189699@1468567152001/A-typical-artificial-neural-network-ANN.png (accedido en Jun. 30, 2022).

[14] Wikipedia, "Archivo:Git-logo.svg," Mayo 2012. [En línea]. Disponible en: https://es.m.wikipedia.org/wiki/Archivo:Git-logo.svg (accedido en Jun. 30, 2022).

[15] D. Robinson, "The Incredible Growth of Python," Septiembre 2017. [En línea]. Disponible en: https://149351115.v2.pressablecdn.com/wp-content/uploads/2017/09/growth_major_languages-1-1400x1200.png (accedido en Jul. 2, 2022).

[16] Wikipedia, "Archivo:JavaScript-logo.png," 2011. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Archivo:JavaScript-logo.png (accedido en Jul. 2, 2022).

[17] Wikipedia, "Archivo:Visual Studio Code 1.35 icon.svg," Junio 2019. [En línea]. Disponible en: https://es.m.wikipedia.org/wiki/Archivo:Visual_Studio_Code_1.35_icon.svg (accedido en Jul. 2, 2022).

[18] M. García, "File:Pandas logo.svg," Octubre 2019. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Pandas_logo.svg (accedido en Jul. 3, 2022).

[19] L. González, "TensorFlow Logo," Mayo 2022. [En línea]. Disponible en: https://aprendeia.com/wp-content/uploads/2022/02/tensorflow-logo.png (accedido en Jul. 3, 2022).

[20] W. Commons, "File:Scikit learn logo small.svg," Septiembre 2012. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Scikit_learn_logo_small.svg (accedido en Jul. 3, 2022).

[21] W. Commons, "File:Node.js logo 2015.svg," 2015. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Node.js_logo_2015.svg (accedido en Jul. 3, 2022).

[22] N. Developer, "React_logo_wordmark," Febrero 2018. [En línea]. Disponible en: https://nightdeveloper.net/react-provider-patron-diseno/react_logo_wordmark/ (accedido en Jul. 3, 2022).

[23] Chart.js, "Stacked Bar Chart." [En línea]. Disponible en: https://www.chartjs.org/docs/latest/samples/bar/stacked.html (accedido en Jul. 3, 2022).

[24] Meta, "React - A JavaScript library for building user interfaces." [En línea]. Disponible en: https://reactjs.org (accedido en Jul. 2, 2022).

[25] S. Ramírez, "FastAPI." [En línea]. Disponible en: https://fastapi.tiangolo.com (accedido en Jul. 2, 2022).

[26] E. Times, "In Formula 1, Engineers Are Essential Members of the Team," Diciembre 2020. [En línea]. Disponible en: https://www.eetimes.eu/in-formula-1-engineers-are-essential-members-of-the-team/ (accedido en Jun. 27, 2022).

[27] Upasana, "Data Science Applications," Abril 2022. [En línea]. Disponible en: https://www.edureka.co/blog/data-science-applications/ (accedido en Jun. 29, 2022).

[28] A. Enyo-one Musa, "Life Cycle of a Data Science Project," Marzo 2021. [En línea]. Disponible en: https://towardsdatascience.com/data-science-101-life-cycle-of-a-data-science-project-86cbc4a2f7f0 (accedido en Jun. 29, 2022).

[29] Wikipedia, "Inteligencia artificial." [En línea]. Disponible en: https://es.wikipedia.org/wiki/Inteligencia_artificial (accedido en Jun. 29, 2022).

[30] D. Alandete, "John McCarthy, el arranque de la inteligencia artificial," Octubre 2011. [En línea]. Disponible en: https://elpais.com/diario/2011/10/27/necrologicas/1319666402_850215.html (accedido en Jun. 29, 2022).

[31] R. España, "Origen del concepto de Inteligencia Artificial," Noviembre 2019. [En línea]. Disponible en: https://agenciab12.com/noticia/origen-concepto-inteligencia-artificial (accedido en Jun. 29, 2022).

[32] IBM, "Deep Blue." [En línea]. Disponible en: https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/ (accedido en Jun. 29, 2022).

[33] Wikipedia, "Aplicaciones de la inteligencia artificial." [En línea]. Disponible en: https://es.wikipedia.org/wiki/Aplicaciones_de_la_inteligencia_artificial (accedido en Jun. 29, 2022).

[34] A. Kongthon, C. Sangkeettrakarn, S. Kongyoung, y C. Haruechaiyasak, "Implementing an online help desk system based on conversational agent," en *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, 2009.

[35] Amazon, "AWS powers F1 Insights." [En línea]. Disponible en: https://aws.amazon.com/f1/ (accedido en Jun. 29, 2022).

[36] Seldon, "Machine Learning Regression Explained," Octubre 2021. [En línea]. Disponible en: https://www.seldon.io/machine-learning-regression-explained (accedido en Jun. 29, 2022).

[37] Wikipedia, "Unsupervised learning." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Machine_learning#Unsupervised_learning (accedido en Jun. 30, 2022).

[38] Wikipedia, "Principal Component Analysis." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Principal_component_analysis (accedido en Jun. 30, 2022).

[39] M. J. Garbade, "Understanding K-means Clustering in Machine Learning," Septiembre 2018. [En línea]. Disponible en: https://towardsdatascience.com/

understanding-k-means-clustering-in-machine-learning-6a6e67336aa1 (accedido en Jun. 30, 2022).

[40] A. Kumar, "Elbow Method vs Silhouette Score – Which is Better?" Noviembre 2021. [En línea]. Disponible en: https://vitalflux.com/elbow-method-silhouette-score-which-better/ (accedido en Jun. 30, 2022).

[41] A. Géron, "The Machine Learning Landscape," en *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow*, Junio 2019, págs. 14–15.

[42] J. McGonagle, G. Shaikouski, y C. Williams, "Backpropagation." [En línea]. Disponible en: https://brilliant.org/wiki/backpropagation/ (accedido en Jul. 2, 2022).

[43] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi *et al.*, "KerasTuner," https://github.com/keras-team/keras-tuner, 2019, (accedido en Jul. 3, 2022).

[44] K. Schwaber y M. Beedle, *Agile Software Development with SCRUM*. Prentice Hall Upper Saddle River, 2002, vol. 1.

[45] R. Hat, "¿Qué es la gestión de la configuración?" Julio 2019. [En línea]. Disponible en: https://www.redhat.com/es/topics/automation/what-is-configuration-management (accedido en Jun. 30, 2022).

[46] Git, "Git." [En línea]. Disponible en: https://git-scm.com (accedido en Jul. 2, 2022).

[47] D. Morán, "F1Data," Junio 2022. [En línea]. Disponible en: https://github.com/DMoran28/F1Data (accedido en Jun. 30, 2022).

[48] Atlassian, "Gitflow Workflow." [En línea]. Disponible en: https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow (accedido en Jun. 30, 2022).

[49] P. S. Foundation, "Python." [En línea]. Disponible en: https://www.python.org (accedido en Jul. 2, 2022).

[50] E. Ecma, "262: Ecmascript language specification," *ECMA (European Association for Standardizing Information and Communication Systems)*, 1999.

[51] S. Tilkov y S. Vinoski, "Node. js: Using javascript to build high-performance network programs," *IEEE Internet Computing*, vol. 14, nº. 6, págs. 80–83, 2010.

[52] Chart.js, "Chart.js." [En línea]. Disponible en: https://www.chartjs.org (accedido en Jul. 2, 2022).

[53] Jupyter, "Project Jupyter." [En línea]. Disponible en: https://jupyter.org (accedido en Jul. 2, 2022).

[54] Microsoft, "Visual Studio Code." [En línea]. Disponible en: https://code.visualstudio.com (accedido en Jul. 2, 2022).

[55] Overleaf, "Overleaf." [En línea]. Disponible en: https://www.overleaf.com/ (accedido en Jul. 2, 2022).

[56] T. L. Project, "LaTeX." [En línea]. Disponible en: https://www.latex-project.org (accedido en Jul. 2, 2022).

[57] P. Schaefer, "FastF1." [En línea]. Disponible en: https://theoehrly.github.io/Fast-F1/ (accedido en Jul. 3, 2022).

[58] A. C. Management, "Pandas." [En línea]. Disponible en: https://pandas.pydata.org (accedido en Jul. 3, 2022).

[59] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, nº. 3, págs. 90–95, 2007.

[60] Google, "TensorFlow." [En línea]. Disponible en: https://www.tensorflow.org (accedido en Jul. 3, 2022).

[61] F. Chollet, "Keras: the Python deep learning API." [En línea]. Disponible en: https://keras.io (accedido en Jul. 3, 2022).

[62] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, y E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, págs. 2825–2830, 2011.

[63] F. Pezoa, J. L. Reutter, F. Suarez, M. Ugarte, y D. Vrgoč, "Foundations of JSON schema," en *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, págs. 263–273.

[64] Wikipedia, "Drag reduction system." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Drag_reduction_system (accedido en Jul. 4, 2022).

[65] ESPN, "Robert Kubica to replace Alfa Romeo's Kimi Raikkonen at Monza," Septiembre 2021. [En línea]. Disponible en: https://www.espn.com/f1/story/_/id/32171965/robert-kubica-replace-alfa-romeo-kimi-raikkonen-monza (accedido en Jul. 4, 2022).

[66] S. Learn, "sklearn.preprocessing.MinMaxScaler." [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html (accedido en Jul. 4, 2022).

[67] S. Learn, "sklearn.decomposition.PCA." [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html (accedido en Jul. 4, 2022).

[68] S. Learn, "sklearn.metrics.silhouette_score." [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html (accedido en Jul. 4, 2022).

[69] Wikipedia, "Rectifier (neural networks)." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Rectifier_(neural_networks) (accedido en Jul. 5, 2022).

[70] Wikipedia, "Sigmoid function." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Sigmoid_function (accedido en Jul. 5, 2022).

[71] Wikipedia, "Dilution (neural networks)." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Dilution_(neural_networks) (accedido en Jul. 5, 2022).

[72] Wikipedia, "Learning rate." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Learning_rate (accedido en Jul. 5, 2022).

[73] J. A. Rodrigo, "Optimización bayesiana de hiperparámetros," Abril 2020. [En línea]. Disponible en: https://www.cienciadedatos.net/documentos/62_optimizacion_bayesiana_hiperparametros.html (accedido en Jul. 5, 2022).

[74] R. Cañadas, "Entropía de Shannon," Agosto 2021. [En línea]. Disponible en: https://abdatum.com/machine-learning/entropia-shannon (accedido en Jul. 5, 2022).

[75] Wikipedia, "Softmax function." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Softmax_function (accedido en Jul. 5, 2022).

[76] Wikipedia, "Business Process Model and Notation." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Business_Process_Model_and_Notation (accedido en Jul. 7, 2022).