UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

"DESARROLLO DE MODELOS BASADOS EN DEEP LEARNING APLICADOS A LA PREDICCIÓN DE RESULTADOS EN CARRERAS DE FÓRMULA 1"

TRABAJO FIN DE GRADO

Febrero -2025

AUTOR: Luna Safi Chekh Zen

DIRECTOR: Antonio Peñalver Benavent

Agradecimientos

Desde que comencé este camino, he aprendido más de lo que imaginaba, y todo ha sido posible gracias al apoyo de las personas que me han acompañado. Este trabajo de fin de grado no es solo el reflejo de mi esfuerzo, sino también de las personas que han estado a mi lado, animándome a seguir adelante.

En primer lugar, quiero agradecer profundamente a mi tutor, Antonio Peñalver, por su constante apoyo, orientación y paciencia a lo largo de este proyecto. Gracias por animarme a seguir adelante con esta idea y por ayudarme a convertirla en algo real, ya que, hacer este trabajo sobre algo que me apasiona mucho como es la Fórmula 1 era algo que me hacía especial ilusión. Su dedicación y valiosos consejos han sido primordiales para el desarrollo de este trabajo. Sin su orientación, no habría llegado hasta aquí.

A mi familia, mis hermanos Dalia y Durri, y en especial a mis padres, les debo todo. Su amor incondicional, sacrificio y confianza han sido mi mayor fuente de motivación. Gracias por estar siempre a mi lado, por su apoyo constante y por creer en mí. No habría logrado llegar hasta aquí sin ellos.

También quiero agradecer a mis amigos y compañeros de clase, quienes han sido una fuente de inspiración, apoyo y ánimo. Muchas veces, sus ideas y ayuda me hicieron superar los obstáculos que surgieron durante mi formación. Gracias por estar siempre dispuestos a ayudarme cuando más lo necesitaba.

A todos ellos, mi eterno agradecimiento. Este logro es también suyo.

Resumen

Este Trabajo de Fin de Grado se centra en la aplicación de técnicas de machine learning para el análisis y la predicción del rendimiento de pilotos y monoplazas en la Fórmula 1, un ámbito caracterizado por el uso intensivo de datos y estrategias avanzadas de ingeniería. El proyecto aborda la construcción y evaluación de cuatro modelos predictivos orientados a diferentes aspectos clave del rendimiento: la predicción de posiciones finales según el circuito, la influencia de las condiciones climáticas, la relación entre las posiciones de clasificación y carrera, y el impacto de las estrategias de neumáticos. Para ello, se han utilizado datos de telemetría, históricos y contextuales obtenidos mediante herramientas específicas.

El desarrollo de este trabajo se fundamenta en el uso de tecnologías avanzadas como Python, junto con bibliotecas y frameworks como PyTorch, Scikit-Learn, y FastAPI, que permitieron la extracción, procesamiento y modelado de datos. Además, se emplearon plataformas como Google Colab y Visual Studio Code para garantizar un flujo de trabajo eficiente. Los modelos desarrollados incluyen enfoques basados en redes neuronales artificiales, desde Perceptrones Multicapa (MLP) hasta arquitecturas avanzadas como LSTM y Transformers, que resultaron ser particularmente útiles en tareas de predicción y análisis de secuencias.

Índice de Contenidos

Índice de Ilustraciones

Índice de Tablas

CÁPITULO 1: INTRODUCCIÓN

1.1 Motivación del proyecto

La Fórmula 1 representa la élite del automovilismo mundial y destaca no solo por el desempeño en la pista, sino también por el uso avanzado de la ingeniería de datos. Los equipos de Fórmula 1 empelan sofisticados sistemas de adquisición de datos y herramientas de análisis para recopilar y examinar grandes cantidades de información en tiempo real y postcarrera, haciendo de este deporte un entorno sumamente intensivo en datos.

En particular, los datos de telemetría juegan un papel fundamental, permitiendo a los equipos monitorear el rendimiento del coche, anticipar su comportamiento bajo diversas condiciones, ajustar estrategias de carrera y optimizar configuraciones vehiculares para mejorar tanto la fiabilidad como el rendimiento. Además, estos datos ayudan a perfeccionar el desempeño de los pilotos durante las sesiones de competencia. Esta capacidad para capturar y analizar datos hace que la Fórmula 1 sea un campo con un enorme potencial para la aplicación de la inteligencia artificial y el machine learning, aunque, sorprendentemente, hay relativamente pocas investigaciones académicas que exploren este ámbito.

Este trabajo de fin de grado se centra en la aplicación de modelos de machine learning para analizar y predecir diversos aspectos del rendimiento en la Fórmula 1. Se representan y discuten cuatro modelos, cada uno orientado a un aspecto particular del rendimiento de los coches y los pilotos, desde la predicción de posiciones en carrera hasta el impacto de las estrategias de neumáticos. El objetivo principal de este proyecto es aprovechar las técnicas de machine learning para transformas los datos de telemetría en conocimientos que contribuyan a la mejora continua de los resultados de carrera.

1.2 Estructura y explicación de la competición de la Fórmula 1

1.2.1 Fórmula 1

La Fórmula 1 (F1) es la categoría más alta del automovilismo mundial reconocida por la Federación Internacional del Automóvil (FIA), caracterizada por ser una competición de élite que combina la excelencia en ingeniería automotriz, estrategia de equipo y habilidades excepcionales de conducción.

Aquí se proporcionará una visión general de los componentes clave que definen cómo funciona la Fórmula 1.

Ilustración 1: Logo oficial de la Fórmula 1[1]

1.2.2 Equipos y Escuderías

Actualmente, la Fórmula 1 está compuesta por 10 equipos, también conocidos como escuderías. Cada equipo, que puede ser una constructora de automóviles o un equipo independiente, fabrica su propio coche de acuerdo con un estricto conjunto de reglas técnicas establecidas por la FIA. Cada escudería compite con dos coches, lo que significa que hay 20 pilotos en total en cada Gran Premio. Algunas de las escuderías más icónicas incluyen Ferrari, Mercedes-AMG Petronas, Red Bull Racing y McLaren, que son reconocidas no solo por su excelencia en la pista sino también por su innovación tecnológica y su capacidad para desarrollar soluciones innovadoras dentro de los límites de los reglamentos.

Cada equipo tiene su propia filosofía de diseño y estrategias de carrera que influyen en la configuración de sus coches y en la forma en la que los pilotos abordan cada Gran Premio.

La competencia entre los equipos no se limita a los pilotos, sino también a los ingenieros, que buscan constantemente optimizar cada componente del coche para obtener la mejor ventaja competitiva.

Ilustración 2: Logo oficial de Scuderia Ferrari [2]

1.2.3 Calificación y Carrera

Los pilotos y los equipos compiten siguiendo un sistema de puntuación a lo largo de la temporada, con el objetivo de ganar tanto el campeonato de pilotos como el campeonato de constructores.

Durante la temporada de Fórmula 1 se celebran en diversas partes del mundo carreras conocidas como **Grandes Premios**, que se llevan a cabo en autódromos y circuitos callejeros. Cada circuito presenta desafíos únicos, desde las estrechas calles de Mónaco hasta las altas velocidades de Monza en Italia, lo que obliga a los equipos a adaptar sus estrategias y configuraciones del coche para cada carrera.

Cada Gran Premio incluye varias etapas: sesiones de práctica, una sesión de clasificación y la carrera principal. Las sesiones de práctica generalmente se realizan los viernes y sábados antes de la carrera, y se llevan a cabo tres sesiones de práctica por cada Gran Premio. Estas sesiones permiten a los equipos probar diferentes configuraciones, adaptarse a las condiciones específicas del circuito y recopilar datos importantes para la carrera.

La sesión de clasificación, que se celebra el sábado, determina el orden de salida en la carrera principal mediante varias rondas en las que los pilotos intentan marcar el mejor tiempo de vuelta posible. La clasificación es muy importante para el rendimiento en la carrera, ya que la posición de salida puede influir significativamente en las oportunidades de adelantar y la estrategia de carrera.

La carrera principal, que se realiza generalmente los domingos, es el evento culminante del fin de semana, donde los pilotos compiten por posiciones, puntos y un lugar en el podio. Las estrategias de carrera, como la elección de neumáticos, la gestión del combustible y los momentos para realizar las paradas en boxes, son determinantes para el éxito en la Fórmula 1. Los equipos deben considerar factores como el desgaste de los neumáticos, las condiciones climáticas y el estado del circuito para tomar decisiones óptima.

Los puntos tras cada carrera principal se otorgan a los primeros 10 pilotos en cruzar la línea de meta, y se otorga 1 adicional al piloto que consiga la vuelta más rápida, siempre y cuando termine dentro del top 10. La distribución de puntos es la siguiente:

- **1ero lugar**: 25 puntos
- **2do lugar**: 18 puntos
- **3er lugar**: 15 puntos
- **4to lugar**: 12 puntos
- **5to lugar**: 10 puntos
- **6to lugar**: 8 puntos
- **7mo lugar**: 6 puntos
- **8vo lugar**: 4 puntos
- **9no lugar**: 2 puntos
- **10mo lugar**: 1 punto

1.2.4 Formato Sprint

Desde 2021, la Fórmula 1 ha introducido un nuevo formato de competición conocido como carreras de sprint, que añade una dimensión adicional a algunos fines de semana de Grandes Premios. Este formato incluye una sesión de clasificación específica para la sprint y una carrera de sprint que se llevan a cabo el viernes y el sábado, respectivamente, por lo que las sesiones de práctica se reducen a sólo una los viernes.

La clasificación de sprint, realizada los viernes, determina el orden de salida para la carrera de sprint del sábado. A diferencia de la carrera que se lleva a cabo el domingo que tiene una duración de 300 kilómetros [3], la carrera de sprint es una competición más corta, de aproximadamente 100 kilómetros o 60 minutos, en la que los pilotos no están obligados a realizar paradas en boxes. Esta carrera proporciona puntos adicionales al campeonato, que se otorgan a los ocho primeros clasificados, siguiendo la siguiente distribución:

- **1er puesto**: 8 puntos
- **2do puesto**: 7 puntos
- **3er puesto**: 6 puntos
- **4to puesto**: 5 puntos
- **5to puesto**: 4 puntos
- **6to puesto**: 3 puntos
- **7mo puesto**: 2 puntos
- **8vo puesto**: 1 punto

Este formato de carrera adicional solo se utiliza en ciertos circuito durante la temporada, y su objetivo es brindar más acción a los aficionados y ofrecer a los equipos una oportunidad extra para sumar puntos.

1.2.5 Reglamentos de la Fórmula 1

La Fórmula 1 no es solo una prueba de velocidad, sino también de cumplimiento de normas estrictas establecidas por la FIA. Estos reglamentos aseguran tanto la seguridad de los pilotos como la igualdad de oportunidades entre los equipos en términos de tecnología y capacidades del coche. Los reglamentos técnicos y deportivos se actualizan frecuentemente para asegurar que el deporte se mantenga competitivo y seguro, lo que obliga a los equipos a innovar constantemente dentro de los límites establecidos.

1.2.5.1 Reglamentos Técnicos

- **Aerodinámica y Estructura del Vehículo:** La FIA implementa pruebas rigurosas para asegurar que las alas delanteras y traseras no excedan los límites de flexibilidad permitidos, evitando ventajas aerodinámicas indebidas. Además, el peso mínimo del monoplaza, incluyendo al piloto y el equipo de grabación de datos, se ha establecido en 798 kg para la temporada 2024 [4].

- **Unidad de Potencia:** Cada piloto puede utilizar hasta cuatro unidades de potencia completas por temporada sin incurrir en penalizaciones. Esto incluye elementos como el motor de combustión interna, el turbo y el MGU-K, que es el componente encargado de recuperar la energía cinética generada durante el frenado y convertirla en energía eléctrica, la cual puede ser utilizada para impulsar el vehículo y mejorar su rendimiento en aceleración.

  Además, se ha incrementado el uso de biocombustibles en la mezcla de combustibles, avanzando hacia una Fórmula 1 más sostenible [4].

- **Neumáticos**: Se ha propuesto la prohibición de las mantas térmicas para neumáticos, permitiendo su uso durante la temporada 2024. Los equipos reciben 13 juegos de neumáticos por fin de semana de carrera, con una asignación para sesiones de práctica, clasificación y carrera [4].

1.2.5.2 Reglamentos Deportivos

- **Penalizaciones**: Las infracciones menores, como exceder el límite de velocidad en el pit lane, resultan en penalizaciones de tiempo o multas económicas. Por otro lado, conductas peligrosas o antideportivas pueden llevar a descalificaciones o suspensiones, dependiendo de la gravedad de la infracción [5]

- **Restricciones de Parc Fermé:** Parc Fermé (o Parque Cerrado) es una zona específica dentro del paddock a la cual los equipos deben llevar sus monoplazas en momentos determinados del fin de semana de carreras. En este lugar [6], comisarios designados por la FIA supervisan los vehículos para asegurarse de que no se realicen modificaciones técnicas con el fin de mejorar el rendimiento.

Ilustración 3: Parc Fermé en la F1

Este tipo de restricción comienzan al inicio de la clasificación y se mantienen hasta el inicio de la carrera, limitando las modificaciones que los equipos pueden realizar en los monoplazas. Estas restricciones tienen como objetivo garantizar que los coches mantengan configuraciones similares desde la clasificación hasta la carrera, preservando la igualdad de condiciones y evitando ajustes significativos que podrían otorgar una ventaja injusta.

- **Procedimientos de Carrera:** El sistema de reducción de arrastre (**DRS**) puede activarse una vuelta después de una salida en carrera, reinicio tras coche de seguridad o bandera roja, lo cual es una vuelta antes que en temporadas anteriores.

  El DRS (***Drag Reduction System*** en inglés) permite a los pilotos reducir la resistencia aerodinámica al abrir una sección del alerón trasero, lo que incrementa la velocidad en recta y facilita los adelantamientos, contribuyendo a una mayor emoción en la carrera. En la ilustración 4 podemos ver un ejemplo de cuando el DRS se activa/desactiva [7].

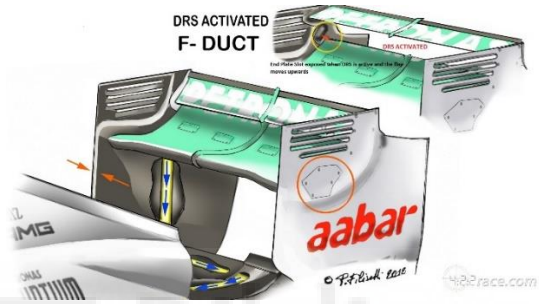

Ilustración 4: DRS abierto vs cerrado

1.2.6 Telemetría dentro de la Fórmula 1

La telemetría es un sistema automatizado que recopila datos de diversos sensores distribuidos en un coche de Fórmula 1 y los transmite en tiempo real a los ingenieros del equipo. Este proceso de recopilación y transmisión de datos permite monitorizar de forma continua el estado y el comportamiento del coche durante las carreras y las sesiones de práctica.

Este sistema abarca una amplia gama de parámetros, que incluyen las condiciones mecánicas y el rendimiento del coche, así como decisiones tácticas y la seguridad del piloto. Estos parámetros permiten a los equipos analizar múltiples aspectos del coche, como la temperatura del motor, la presión de los frenos, la velocidad, la aceleración, y el desgaste de los neumáticos, entre otros. Gracias a estos datos, los ingenieros pueden tomar decisiones rápidas y optimizar las configuraciones del coche, lo cual es esencial para maximizar el rendimiento y garantizar la seguridad del piloto.

Durante las sesiones de clasificación, la telemetría juega un papel muy importante, ya que cada vuelta cuenta para definir la posición de salida. Durante una vuelta rápida, los sensores del coche recopilan información sobre la velocidad en las rectas, el ángulo de dirección en las curvas, la presión ejercida sobre los frenos y el nivel de agarre de los neumáticos. Esta información es transmitida instantáneamente al equipo, lo que permite ajustar la configuración del vehículo en tiempo real para mejorar el rendimiento en las siguientes vueltas.

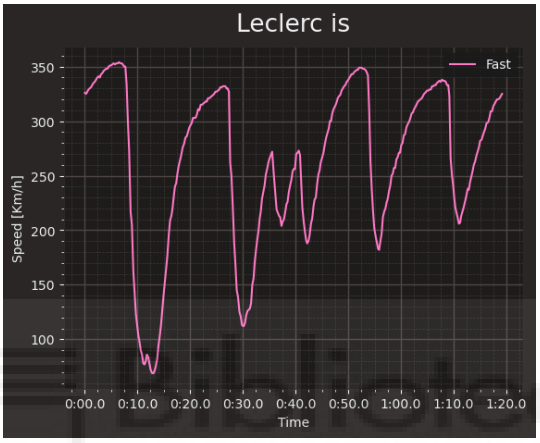

Ilustración 5: Telemetría asociada a un piloto durante una vuelta en una clasificación

Además, este sistema es vital para la seguridad, ya que permite a los equipos detectar problemas mecánicos o fallas críticas antes de que puedan causar accidentes o daños mayores. Los sensores pueden alertar sobre temperaturas excesivas, fallos de presión de los neumáticos o anomalías en el sistema de frenos, lo que permite tomar medidas preventivas para evitar incidentes.

En resumen, la telemetría permite a los equipos de Fórmula 1 hacer uso de la ciencia de datos para tomar decisiones informadas y estratégicas, las cuales pueden marcar la diferencia entre ganar o perder en el circuito. Su evolución continua promete seguir revolucionando la manera en que los equipos abordan cada aspecto del automovilismo, permitiendo una mejora contante en el rendimiento, la eficiencia y la seguridad.

CÁPITULO 2:
ESTADO DE LA CUESTIÓN

2.1 Inteligencia Artificial

2.1.1 Introducción a la Inteligencia Artificial

La **inteligencia artificial (IA)** es una rama de la informática que se dedica al desarrollo de sistemas capaces de realizar tareas que normalmente requieren inteligencia humana, tales como el razonamiento, el aprendizaje, la percepción y la toma de decisiones. La idea de crear máquina que puedan simular procesos cognitivos humanos no es nueva, y sus primeras manifestaciones se remontan a mediados del siglo XX.

La IA como disciplina formal tuvo sus orígenes en la década de 1959, cuando **Alan Turing** planteó la pregunta fundamental: "*¿Pueden las máquina pensar?*" [8]. En 1956, la conferencia de **Darthmouth**, liderada por **John McCarthy**, marcó el inicio oficial de la investigación en IA, sentando las bases para lo que sería un campo en constante evolución.

A lo largo de su desarrollo, la IA ha atravesado diferentes ciclos de auge y declive, conocidos como los "veranos" e "inviernos" de la IA. Estos ciclos reflejan momentos de grandes avances tecnológicos y expectativas no cumplidas, que dieron lugar a períodos de escepticismo y falta de inversión [8]. Actualmente, la IA vive uno de sus momentos de mayor auge, debido al avance en capacidad de procesamiento, la disponibilidad de grandes volúmenes de datos y las mejoras en los algoritmos.

La IA moderna se ha convertido en una tecnología clave en múltiples sectores, proporcionando capacidades para la **automatización de procesos, análisis de grandes volúmenes de datos,** y **mejora en la toma de decisiones** en ámbitos como la medicina, la industria, el comercio y más.

2.1.2 Aplicaciones de la Inteligencia Artificial

La IA está presente en muchos aspectos de la vida cotidiana y ha tenido un impacto significativo en diversas industrias y sectores. A continuación, se describen algunas de las aplicaciones más relevantes:

- **Salud:** En el ámbito de la salud, la IA se utiliza para **diagnosticar enfermedades** mediante el análisis de imágenes médicas y para desarrollar nuevos tratamientos. Sistemas como **Watson Health** de IBM son capaces de analizar diagnósticos precisos, ayudando a los médicos a tomar decisiones informadas [9]. Además, se están desarrollando **robots quirúrgicos** asistidos por IA que permiten realizar cirugías con una mayor precisión.

- **Finanzas:** En el sector financiero, la IA juega un papel importante en la **detección de fraudes**, el análisis de riesgos y la **automatización de servicios financieros.** Los algoritmos de aprendizaje automático se empelan para identificar patrones inusuales en transacciones y prevenir actividades fraudulentas. También se utilizan para proporcionar asesoramientos financiero personalizado a los clientes.

- **Automatización y Robótica:** En la manufactura y la robótica, la IA se utiliza para automatizar procesos de producción y mejorar la eficiencia. Robots inteligentes pueden realizar tareas repetitivas con gran precisión, y los algoritmos de la IA permiten optimizar la planificación de la producción y el mantenimiento preventivo de las máquinas.

- **Transporte:** La IA es esencial en el desarrollo de los **vehículos autónomos**. Empresas como **Tesla** y **Waymo** han desarrollado sistemas que permiten que los automóviles se conduzcan sin intervención humana. Estos vehículos utilizan una combinación de **visión por computadora, sensores** y **redes neuronales** para analizar el entorno, tomar decisiones y navegar de manera segura [9].

- **Atención al Cliente:** Los **chatbots** y **asistentes virtuales**, como **Siri** o **Alexa**, utilizan IA para interactuar con los usuarios de una forma más natural y resolver sus

consultas. Estas tecnologías han mejorado significativamente la experiencia del usuario, ofreciendo respuestas inmediatas y soporte en cualquier momento.

- **Comercio Electrónico:** En el ámbito del comercio, la IA se utiliza para proporcionar recomendaciones de productos basadas en el comportamiento de compra de los usuarios. Plataformas como **Amazon** emplean algoritmos de aprendizaje automático para garantizar el historial de búsqueda y compras de los clientes, y así ofrecer sugerencias personalizadas, mejorando la experiencia del cliente y aumentando ventas.

- **Educación:** En el sector educativo, la IA está revolucionando el aprendizaje mediante **tutorías inteligentes**, personalización de contenidos y **evaluación automatizada**. Los sistemas de aprendizaje adaptativo permiten a los estudiantes aprender a su propio ritmo, ajustando los materiales según sus necesidades y progresos.

2.1.3 Ramas de la Inteligencia Artificial

La inteligencia artificial se divide en varias subramas, cada una de las cuales abordan aspectos específicos de la construcción de sistemas inteligentes. Las principales ramas de la IA son las siguientes:

1. **Machine Learning (Aprendizaje Automático):** El **Machine Learning** (**ML**) es una de las ramas más importantes de la inteligencia artificial. Se basa en la capacidad de los algoritmos para **aprender a partir de datos** y mejorar su rendimiento sin necesidad de ser programados explícitamente para cada terea [10].

2. **Deep Learning (Aprendizaje Profundo):** El **Deep Learning (DL)** es una subrama del aprendizaje automático que utiliza redes neuronales profundas para modelar relaciones complejas en los datos. Estas redes están inspiradas en la estructura del cerebro humano y permiten realizar tareas como el reconocimiento de voz, el procesamiento del lenguaje natural y el análisis de imágenes.

Una de las aplicaciones más destacadas del aprendizaje profundo es el **reconocimiento facial**, que se utiliza tanto en dispositivos móviles como en sistemas de seguridad [11]. Además, las redes neuronales **convolucionales (CNN)** se emplean para la **clasificación de imágenes**, mientras que las **redes neuronales recurrentes (RNN)** son útiles para el análisis de secuencias de datos, como el texto o las series temporales

3. **Natural Language Processing (Procesamiento del Lenguaje Natural):** El **natural language processing (NLP)** permite a las máquinas comprender e interactuar con el lenguaje humano de una manera significativa. **ChatGPT** es un ejemplo de cómo el NLP se puede utilizar para **generar respuestas coherentes** y resolver problemas planteados por los usuarios. La capacidad de comprender y generar texto natural tiene aplicaciones en la **traducción automática,** la **clasificación de sentimientos** y los **asistentes virtuales** [11].

4. **Visión por Computadora:** La **visión por computadora** es la rama de la IA que se ocupa de dotar a las máquinas de la capacidad de "ver" y **entender el conocimiento de las imágenes** y videos. Se utiliza en aplicaciones que van desde el reconocimiento facial hasta la inspección de calidad en la producción industrial. Este campo combina técnicas de **análisis de imagen** con algoritmos de aprendizaje profundo para extraer información significativa a partir de entradas visuales.

5. **Robótica:** La **robótica** es otro subcampo de la IA que combina hardware y software para desarrollar **máquinas capaces de realizar tareas físicas**. Los robots industriales, los drones y los vehículos autónomos son ejemplos de cómo la robótica y la IA pueden trabajar en conjunto para automatizar tareas complejas. La robótica emplea técnicas de control, visión por computadora y aprendizaje automático para mejorar la interacción de los robots con su entorno [11].

2.2 Machine Learning

El Aprendizaje Automático (Machine Learning, ML) es una subdisciplina de la IA que permite a los sistemas aprender de los datos y mejorar su desempeño sin ser programados explícitamente. Se centra en el desarrollo de algoritmos que identifican patrones en los datos y utilizan estos patrones para realizar predicciones o decisiones automatizadas [12]. El ML ha revolucionado diferentes campos, incluyendo la medicina, las finanzas, el comercio, y la industria, ya que permite desarrollar soluciones adaptativas y más precisas.

2.2.1 Tipos de Machine Learning

El aprendizaje automático se clasifica principalmente en tres categorías: **aprendizaje supervisado, aprendizaje no supervisado,** y **aprendizaje por refuerzo.** Cada tipo tiene enfoques específicos y algoritmos que se utilizan para diferentes propósitos, como la clasificación, agrupamiento y reducción de dimensionalidad.

2.2.1.1 Aprendizaje supervisado

En el aprendizaje supervisado, los algoritmos se entrenan con **datos etiquetados**, donde el conjunto de datos incluye entradas y salidas conocidas. El objetivo es encontrar una función que pueda mapear las entradas a las salidas correctas para nuevos datos. Los algoritmos se dividen principalmente en dos tipos: **clasificación y regresión.**

- **Clasificación:** En problemas de clasificación, los algoritmos aprenden a asignar etiquetas a diferentes clases. Ejemplos de algoritmos incluyen:

    - **Máquinas de Vectores de Soporte (SVM):** Eficaz para clasificación binaria y multiclase, busca el hiperplano que mejor separa las clases [13].
    - **Árboles de Decisión:** Algoritmo basado en reglas que divide los datos en subconjuntos basados en decisiones lógicas, fácilmente interpretable.
    - **Redes Neuronales Artificiales (ANN):** Modelos inspirados en el cerebro humano, que son capaces de capturar relaciones complejas en los datos mediante capas de neuronas interconectadas y que se utilizan en una variedad de aplicaciones, como reconocimiento de patrones e imagen [13]. Dentro de este tipo se incluyen las **Redes Neuronales Perceptrón Multicapa (MLP)**,

que son capaces de aprender patrones no lineales gracias a la presencia de capas ocultas, y las **Redes Transformer**, que son especialmente efectivos para procesar secuencias de datos, ya que permiten el aprendizaje contextual con mecanismos de atención.

- **Regresión:** En problemas de regresión, los algoritmos intentan predecir un valor continuo.
  - **Regresión Lineal:** Se utiliza para predecir valores continuos basados en la relación entre variables independientes y dependientes.
  - **Regresión Logística:** Aunque su nombre sugiere regresión, se utiliza principalmente para problemas de clasificación binaria, prediciendo la probabilidad de pertenecer a una clase.
  - **Regresión con Redes Neuronales:** ANN también se puede usar para problemas de regresión cuando los datos son complejos y no lineales.

2.2.1.2 Aprendizaje no supervisado

En el aprendizaje no supervisado, los algoritmos trabajan con **datos no etiquetados** y buscan patrones o estructuras ocultas sin conocer la salida correcta de antemano. Esto permite descubrir características comunes en los datos, y se utilizan técnicas de **agrupamiento**, **reducción de dimensionalidad** y **detección de anomalías**.

- **Agrupamiento (Clustering):** Algoritmos que dividen los datos en grupos según sus similitudes. Algunos ejemplos son:
  - **K-means:** Algoritmo de agrupamiento que clasifica los datos en "K" grupos basándose en sus características [14]. Es útil para la segmentación de clientes y otros tipos de análisis exploratorios.

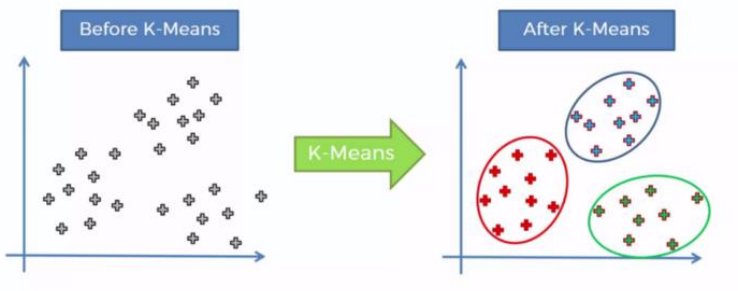

Ilustración 6: Ejemplo de clustering con K-Means

- **Algoritmo de Agrupamiento Jerárquico:** Agrupa los datos en una jerarquía de clústeres que se visualiza a través de un **dendrograma**, permitiendo identificar relaciones entre los grupos de manera jerárquica.

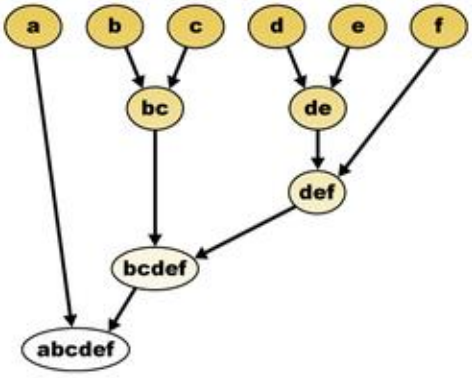

Ilustración 7: Diagrama de agrupación jerárquica [15]

- **DBSCAN:** Algoritmo basado en la densidad que identifica clústeres de diferentes formas y es útil para detectar datos atípicos.

- **Redes Neuronales Autoorganizadas (SOM):** Redes que agrupan los datos en mapas visuales de dos dimensiones, facilitando la comprensión de patrones complejos.

2.2.1.3 Aprendizaje por refuerzo

El aprendizaje por refuerzo involucra a un **agente** que aprende interactuando con un entorno. La idea principal es maximizar una **recompensa acumulada** a lo largo del tiempo mediante la experimentación y el ajuste de las acciones del agente.

- **Q-learning:** Algoritmo que busca aprender una política óptima para maximizar la recompensa futura acumulada. Se usa ampliamente en juegos y optimización de sistemas.
- **Rede Neuronales Actor-Crítico**: Un enfoque híbrido que combina redes para la selección de acciones (actor) y redes para evaluar el valor de dichas acciones (crítico).

- **Redes Neuronales Recurrentes (RNN)**: Especialmente útiles en aprendizaje por refuerzo para tareas que implican secuencias temporales, ya que pueden mantener información sobre estados previos y, por lo tanto, tomar decisiones basadas en la experiencia pasada.
- **Redes LSTM (Long Short-Term Memory):** Un tipo específico de RNN (Red Neuronal Recurrente) que maneja mejor la dependencia a largo plazo, evitando problemas de desvanecimiento del gradiente y siendo muy útil en aplicaciones como la predicción de series temporales y el análisis secuencial en entornos dinámicos.

2.2.2 Técnicas de Aprendizaje

Existen diversas técnicas que complementan los diferentes tipos de aprendizaje automático para mejorar la calidad y eficiencia de los modelos. Algunas de las técnicas más comunes son:

- **Árboles de Decisión:** Utilizados tanto para **clasificación** como una **regresión**, los árboles de decisión dividen los datos en subconjuntos con base en reglas lógicas y permiten crear modelos simples y explicables. Estos se pueden combinar para formas modelos más complejos como los **bosques aleatorios.**

- **Redes Bayesianas:** Modelos gráficos probabilísticos que representan un conjunto de variables y sus relaciones de dependencia condicional mediante un grafo dirigido. Las redes bayesianas son particularmente útiles en el análisis de riesgos y la toma de decisiones bajo incertidumbre.

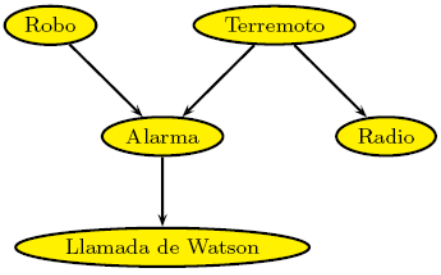

Ilustración 8: Red Bayesiana del ejemplo robo o terremoto

El diagrama la ilustración 8 [16] muestra una **Red Bayesiana** que modela la activación de una alarma ante un **robo** o un **terremoto**, donde ambos eventos pueden influir en la alarma. Si la alarma suena, puede hacer que **Watson llame**, mientras que un **terremoto** también puede generar alertas en la **radio**. Las flechas indican relaciones de dependencia condicional, permitiendo calcular probabilidades, como la posibilidad de que haya ocurrido un robo si la alarma se activa.

- **Ensamblado (Ensemble Learning):** Combina múltiples modelos individuales para mejorar el rendimiento. Algunos ejemplos incluyen

  - **Bosques Aleatorios (Random Forest):** Combinación de varios árboles de decisión, donde cada árbol contribuye con un voto para determinar la predicción final [17].

  - **Boosting:** Como **AdaBoost** y **Gradient Boosting**, son técnicas que ajustan secuencialmente modelos débiles, mejorando iterativamente la calidad de la predicción.

- **Reducción de Dimensionalidad:** Técnicas como **PCA** y el **Análisis Discriminante Lineal (LDA)** se usan para reducir la cantidad de variables en el modelo, lo cual facilita el cálculo, mejora la interpretabilidad y elimina ruido innecesario.

Estas técnicas y algoritmos permiten desarrollar modelos de aprendizaje automático más precisos, robustas y eficientes, facilitando así la implementación de soluciones inteligentes en una amplia variedad de aplicaciones.

2.3 Deep Learning

Como se ha mencionado anteriormente, el **Deep Learning** es una subrama del aprendizaje automático que se centra en el uso de redes neuronales profundas para modelar patrones complejos y aprender representaciones de alto nivel de los datos. A diferencia de otros enfoques del aprendizaje automático, que pueden requerir una elaboración exhaustiva de las características, el deep learning tiene la capacidad de aprender directamente de los datos sin intervención humana significativa, gracias a la estructura jerárquica de sus redes neuronales.

Este tipo de aprendizaje ha cobrado gran relevancia en los últimos años debido a su eficacia en tareas donde se requiere una comprensión detallada y compleja, como el reconocimiento de imágenes, el procesamiento del lenguaje natural y la generación automática de contenido. Las redes profundas utilizan múltiples capas de neuronas para extraer características a distintos niveles de abstracción, lo que les permite descubrir patrones ocultos que antes eran difíciles de identificar con otros métodos.

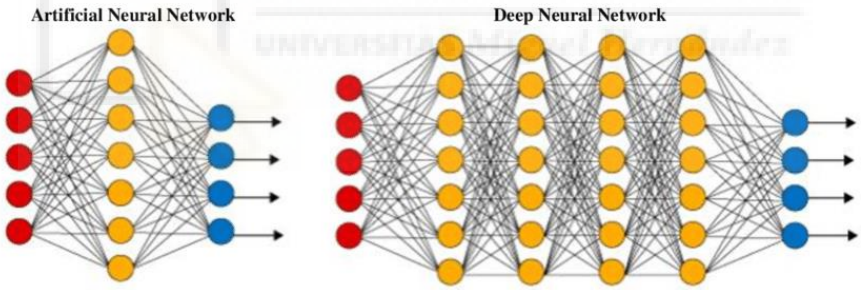

Ilustración 9: Comparación ANN con una DNN [18]

El Deep Learning se basa en redes neuronales que pueden tener decenas o incluso miles de capas, denominadas “profundas” debido a su capacidad para profundizar en el entendimiento de los datos y sus relaciones.

2.4 Redes Neuronales Artificiales

Las Redes Neuronales Artificiales (RNA)/Artificial Neural Networks (ANN) son modelos computacionales que se inspiran en el funcionamiento del cerebro humano para resolver problemas complejos de predicción, clasificación y reconocimiento de patrones. Estas redes simulan la forma en que las neuronas biológicas procesan la información mediante la conexión de múltiples unidades o nodos, denominadas **neuronas artificiales**, que trabajan en conjunto para identificar patrones ocultos y relaciones en los datos.

Las RNAs son particularmente útiles en contextos donde las relaciones entre los datos son difíciles de modelar mediante enfoques tradicionales, como la regresión lineal. Las redes pueden aprender a partir de ejemplos, lo que significa que se entrenan utilizando grandes conjuntos de datos para ajustar sus parámetros internos, conocidos como **pesos**, de modo que se minimicen los errores en las predicciones.

Una de las principales ventajas de las RNAs es su capacidad para **generalizar**, es decir, aprender reglas a partir de datos de entrenamiento y aplicarlas a nuevas situaciones. Esta capacidad las hace muy útiles en una variedad de aplicaciones que incluyen el reconocimiento de voz, la visión por computadora, la detección de fraudes y la predicción de tendencias en el mercado financiero.

Este tipo de redes no requieren una elaboración exhaustiva de características como algunos otros modelos, ya que pueden aprender representaciones diferente de los datos en bruto.

2.4.1 Estructura de una Red Neuronal Artificial

La estructura de una Red Neuronal Artificial está compuesta por una serie de neuronas agrupadas en capas que trabajan de manera colaborativa para procesar y transformas los datos de entrada en resultados útiles. Las RNAs suelen estar organizadas en las siguientes capas principales:

1. **Capa de Entrada (Input Layer):** La capa de entrada es la primera capa de la red y tiene la función de recibir los datos de entrada. Cada nodo en esta capa

representa una característica o variable del conjunto de datos. No realiza ningún tipo de procesamiento; simplemente transmite la información a las capas siguientes.

2. **Capas Ocultas (Hidden Layers):** Las capas ocultas son las encargadas de realizar la mayor parte del procesamiento de los datos. Estas capas están formadas por neuronas artificiales que aplican funciones de activación no lineales para aprender características complejas y patrones ocultos en los datos. Las redes profundas pueden contener múltiples capas ocultas, lo que les permite aprender representaciones jerárquicas y captar relaciones de alta complejidad.

3. **Capa de Salida (Output Layer):** La capa de salida es la última capa de la red y proporciona el resultado final del procesamiento. El número de nodos en la capa de salida depende del tipo de problema que se esté resolviendo. Por ejemplo, para un problema de clasificación binaria, habrá un único nodo que indique la probabilidad de pertenecer a una de las dos clases, mientras que para una clasificación multiclase, habrá tantos nodos como clases posibles.

4. **Pesos y Biases:** Las conexiones entre las neuronas de una RNA están ponderadas con valores conocidos como **pesos**. Estos pesos determinan la importancia de la señal que se transmite entre dos neuronas. Además, cada neurona tiene un valor adicional llamado **bias** que ayuda a ajustar la salida de la neurona. Durante el entrenamiento, los pesos y biases se ajustan para minimizar el error del modelo.

5. **Funciones de Activación:** Las funciones de activación introducen no linealidad en la red, lo cual es fundamental para que la RNA pueda aprender patrones complejos. Algunas de las funciones de activación más comunes son la **ReLU (Rectified Linear Unit),** la **sigmoide**, y la **tangente hiperbólica (tanh)**. Estas funciones determinan si una neurona debe activarse o no en función de la entrada recibida.

La ilustración siguiente [19] representa una **red neuronal artificial** con tres capas: **entrada, oculta y salida**.

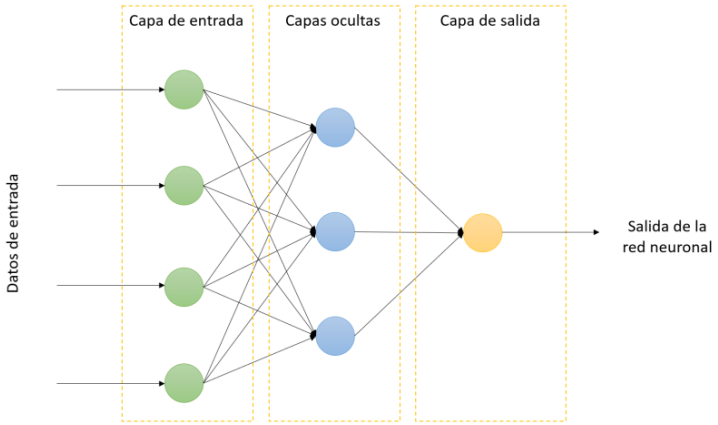

Ilustración 10: Arquitectura de una RNA

Las RNAs aprenden ajustando los pesos y biases mediante un proceso iterativo llamado **entrenamiento,** que se basa en minimizar una función de pérdida que mide como de lejos está la predicción del modelo respecto al valor real. Este proceso es clave para garantizar que la red pueda generalizar adecuadamente a nuevos datos.

2.4.2 Tipos de Red Neuronales

Existen diferentes tipos de redes neuronales, cada una adaptada a resolver distintos problemas y con estructuras específicas. Aunque hay muchas variantes, en este trabajo nos centraremos en tres tipos de redes neuronales que fueron empleadas en este trabajo: el **Perceptrón Multicapa (MLP)**, la **Red Neuronal Recurrente (RNN)**, la **Red LSTM**, **y** las **Redes Transformer**.

2.4.2.1 Perceptrón Multicapa (MLP)

El **MLP** es una extensión del perceptrón simple y constituye una de las arquitecturas más básicas en el aprendizaje profundo. Un MLP está compuesto por una capa de entrada, una o más capas ocultas, y una capa de salida. Las capas ocultas permiten al MLP aprender patrones no lineales complejos, lo que lo hace apto para tareas como la clasificación y la regresión.

Arquitectura del MLP:

- Capa de entrada: Esta recibe los datos de entrada y cada neurona representara una característica del conjunto de datos.

- Capas ocultas: Tienen como función procesar la información aplicando funciones de activación no lineales. Las capas ocultas suelen estar conectadas del todo, es decir, cada neurona de una capa está conectada con todas las neuronas de la siguiente capa.

- Capa de salida: Proporciona la predicción final, cuyo número de neuronas depende del tipo de problema (regresión, clasificación binaria, multiclase).

- Las **funciones de activación** comunes en las capas ocultas son **ReLU** y **sigmoide**, mientras que la capa de salida podría usar una función **softmax** en problemas de clasificación [20].

2.4.2.2 Red Neuronal Recurrente

Las RNN son un tipo de red neuronal diseñada concretamente para trabajar con datos secuenciales, como son las series temporales o el lenguaje natural. A diferencia de los MLP, las RNN tiene **conexiones recurrentes** que permiten que la información persista y se transmita de una iteración a la siguiente, proporcionando a la red un tipo de "memoria" que facilita el modelado de relaciones temporales [21].

Arquitectura de la RNN:

La arquitectura de una RNN se caracteriza por tener **celdas recurrentes** que permiten la retroalimentación de la información. Esto quiere decir que la salida de una neurona en una capa es capaz de convertirse en la entrada para la misma capa en la siguiente iteración de tiempo.

Las **RNN estándar** tienen dificultades para aprender dependencias a largo debido al problema del **gradiente que se desvanece**, que afecta a la capacidad de la red para actualizar adecuadamente los pesos en capas profundas.

En las funciones de activación comunes nos encontramos tales como **tanh** y **sigmoide**, que permiten regular la activación de las neuronas a lo largo del tiempo, aunque estas funciones tambien contribuyen al problema del gradiente [22].

2.4.2.3 Red LSTM (Long Short-Term Memory)

La Red LSTM es una versión mejorada de las RNN diseñada para resolver el problema del gradiente que se desvanece. Las LST tienen una arquitectura más compleja la cual incluye **celdas de memoria**, además de puertas de entrada, salida y olvido, que controlan el flujo de información y permiten que la red mantenga y actualice la información durante largos periodos de tiempo [23].

Arquitectura de la LSTM:
La arquitectura de una LSTM consiste en **celdas de memoria** que almacenan información relevante durante varios intervalos de tiempo. Estas celdas están equipadas con **tres tipos de puertas**:

- **Puerta de Olvida (Forget Gate):** Decide cuánta información de la celda de memoria debe olvidarse.
- **Puerta de Entrada (Input Gate):** Determina cuánta información nueva debe almacenarse en la celda de memoria.
- **Puerta de Salida (Output Gate):** Decide qué parte de la información en la celda de memoria se empelará como salida en la iteración actual.

Las LSTM son capaces de aprender dependencias tanto a corto como a largo plazo, lo que las hace especialmente aptas para tareas de secuencias largas y complejas, como **predicción de series temporales** y **traducción automática** [24].

2.4.2.4 Redes Transformer

Las redes Transformer son un tipo de arquitectura de redes neuronales artificiales que han cambiado radicalmente el campo del procesamiento del lenguaje natural (NLP) y otras aplicaciones que trabajan con secuencias de datos. Fueron presentadas por primera vez en 2017 por Vaswani y su equipo, y rápidamente superaron muchas de las limitaciones de arquitecturas anteriores, como las Redes Neuronales Recurrentes (RNN) y las Long Short-Term Memory (LSTM). Esto permitió procesar secuencias de una forma mucho más eficiente y efectiva. Desde entonces, los Transformers se han convertido en la arquitectura preferida para tareas que van desde la traducción automática hasta la generación de texto, la clasificación de secuencias, e incluso aplicaciones en visión por computadora y modelado de proteínas.

El componente clave que hace a los Transformers tan únicos es el uso del **mecanismo de atención**. Este mecanismo le permite al modelo enfocarse de manera selectiva en diferentes partes de la entrada, algo muy útil cuando se trabaja con secuencias largas. A diferencia de las RNN, que procesan la información de manera secuencial (es decir, palabra por palabra o elemento por elemento), los Transformers son capaces de analizar toda la secuencia simultáneamente. Esto se logra gracias a la **atención auto-regresiva** (self-attention), que asigna distintos pesos a cada palabra o elemento de la secuencia, para determinar cuáles partes son más importantes al generar la salida. Este enfoque no solo acelera considerablemente el procesamiento, sino que también permite captar mejor las relaciones a largo plazo dentro de las secuencias.

Arquitectura de un Transformer:

La arquitectura básica de un Transformer tiene dos partes principales: el **codificador** (encoder) y el **decodificador** (decoder). En tareas como la traducción de texto, el codificador se encarga de convertir la secuencia de entrada (por ejemplo, una oración en inglés) en una representación interna, mientras que el decodificador usa esa representación para generar la salida correspondiente (la traducción en otro idioma). Cada uno de estos componentes está formado por varias capas que contienen dos subcapas: una capa de atención y una capa de red neuronal feedforward. Además, cada subcapa está acompañada de mecanismos de **normalización** y **residual**, que ayudan a estabilizar el aprendizaje y mejorar la eficiencia.

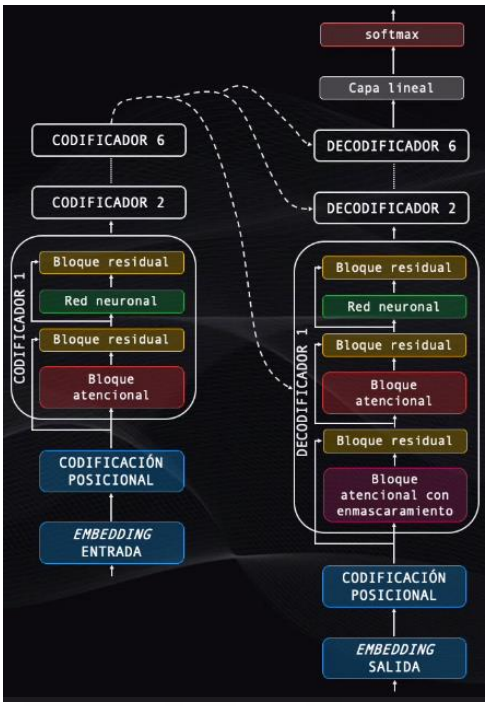

Ilustración 11: Diagrama de bloques general de una Red Transformer [25]

En el **codificador**, se utiliza la **atención auto-regresiva** para analizar la secuencia de entrada y determinar qué partes de esa secuencia son más importantes. Este proceso se realiza en paralelo, lo que permite al modelo entender simultáneamente las relaciones entre todas las palabras, sin necesidad de pasar secuencialmente por cada una de ellas, como ocurriría en las **RNN**. Por otro lado, el **decodificador** utiliza una combinación de atención sobre la secuencia generada hasta el momento y atención sobre la representación obtenida por el codificador, lo cual permite generar la secuencia de salida de forma coherente y contextual.

Una de las principales razones por las que los Transformers han tenido tanto éxito es la capacidad de paralelizar el procesamiento de secuencias. Esto hace posible entrenar modelos mucho más grandes y con muchos más datos, lo que ha llevado al desarrollo de modelos como **GPT-3** y **BERT**, que son capaces de generar y comprender texto a un nivel casi humano. Además de su uso en **NLP**, los **Transformers** han empezado a aplicarse en otros campos, como la visión por computadora (Vision Transformers o ViT), donde también están demostrando ser muy efectivos al procesar imágenes de manera similar a como procesan el texto.

Estas cuatro arquitecturas han sido fundamentales en la construcción de los modelos de predicción de carrera desarrollados en este proyecto, cada una aportando ventajas concretas dependiendo del tipo de datos y el problema a resolver.

2.4.3 Entrenamiento de una Red Neuronal

El entrenamiento de una Red Neuronal Artificial (RNA) es un proceso imprescindible que permite a la red aprender patrones a partir de datos, ajustando sus parámetros internos (pesos y biases) para mejorar su capacidad predictiva. Este proceso se realiza mediante una serie de pasos iterativos, desde la preparación de los datos hasta la evaluación del rendimiento final del modelo. A continuación, se detallan las etapas clave del entrenamiento de una RNA:

1. **Preparación de los Datos**

   Antes de iniciar el entrenamiento, es importante preparar adecuadamente los datos para garantizar que la red aprenda de forma efectiva:

   - **Recolección y Limpieza de Datos:** Es fundamental contar con un conjunto de datos representativo y de alta calidad. Esto implica la eliminación de valores atípicos, el manejo de datos faltantes y la corrección de errores para evitar que la red aprenda patrones incorrecto o sesgados [26].

   - **División del Conjunto de Datos:** El conjunto de datos se divide generalmente en tres subconjuntos: **entrenamiento, validación,** y **prueba**. El conjunto de entrenamiento se utiliza para ajustar los parámetros de la red, el de validación para ajustar hiperparámetros y prevenir el sobreajuste, y el conjunto de prueba para evaluar el rendimiento del modelo en datos no vistos.

   - **Normalización y Escalado:** La normalización transforma las características numéricas para que tengan una escala similar, facilitando la convergencia durante el entrenamiento y evitando que algunas variables dominen a otras debido a su magnitud.

2. **Inicialización de Pesos y Biases**

   La iniciación de los pesos y biases es un paso crítico que afecta directamente la velocidad y eficacia del entrenamiento:

   - **Inicialización Aleatoria:** Los pesos se inicializan con valores aleatorios pequeños para romper la simetría y permitir que las neuronas aprendan diferentes características.

   - **Técnicas de Inicialización Avanzadas:** Método como la inicialización de **He** o **Xavier** ajustan los valores iniciales en función del número de neuronas en las capas, mejorando la estabilidad y la velocidad de convergencia.

3. **Propagación Hacia Adelante (Forward Propagation)**

   Durante la propagación hacia adelante, los datos se transmiten a través de la red para generar una salida:

   - **Cálculo de Salidas:** Cada neurona procesa la entrada aplicando una función de activación a la suma ponderada de sus entradas, generando una salida que se propaga a la siguiente capa.
   - **Funciones de Activación:** Las funciones de activación, como **ReLU, sigmoide** y **tangente hiperbólica (tanh)**, introducen no linealidad, lo cual es esencial para que la red pueda aprender relaciones complejas entre las entradas y salidas [27].

4. **Cálculo de la Función de Pérdida**

   La función de pérdida cuantifica la diferencia entre las predicciones de la red y los valores reales esperados. Minimizar esta pérdida es el objetico principal del entrenamiento:

   - **Funciones de Pérdida Comunes:** En problemas de regresión se suele utilizar el **Error Cuadrático Medio (MSE)**, mientras que en problemas de clasificación se emplea la **Entropía Cruzada**.

5. **Propagación Hacia Atrás (Backpropagation)**

   La propagación hacia atrás es el proceso mediante el cual se ajustan los pesos y biases para minimizar la pérdida:

   - **Cálculo de Gradientes:** Se utiliza el algoritmo de backpropagation para calcular los gradientes de la función de pérdida con respecto a cada peso, utilizando el cálculo de derivadas parciales.

   - **Actualización de Pesos:** Los pesos se actualizan en la dirección opuesta al gradiente (descenso de gradiente) para reducir la pérdida, ajustando los parámetros de la red en cada iteración.

6. **Optimización**

   Los optimizadores son algoritmos que controlan cómo se actualizan los pesos y biases durante el entrenamiento:

   - **Descenso de Gradiente Estocástico (SGD):** Este algoritmo actualiza los parámetros utilizando un subconjunto aleatorio de datos en cada iteración, lo cual puede acelerar el entrenamiento al reducir la carga computacional [28].

   - **Optimizadores Avanzados:** Algoritmos como **Adam** y **RMSprop** adaptan dinámicamente las tasas de aprendizaje para cada parámetro, mejorando la eficiencia y la velocidad de convergencia.

7. **Regularización**

   Para evitar el sobreajuste y garantizar que el modelo generalice bien a nuevos datos, se utilizan técnicas de regularización:

   - **L1 y L2:** Estas técnicas agregan una penalización a la función de pérdida proporcional a la magnitud de los pesos, incentivando modelos más simples.

- **Dropout:** Durante el entrenamiento, **dropout** desactiva aleatoriamente algunas neuronas, lo cual evita que la red dependa demasiado de ciertas neuronas y mejora la robustes del modelo.

8. **Validación y Ajuste de Hiperparámetros**

    - **Ajuste de Hiperparámetros:** Los hiperparámetros, como la tasa de aprendizaje, el número de capas y neuronas, y las funciones de activación, se ajustan para optimizar el rendimiento del modelo.

    - **Estrategias de Búsqueda:** Se emplean método como la **búsqueda en cuadrícula** o la **búsqueda aleatoria** para encontrar combinaciones óptimas de hiperparámetros.

9. **Evaluación Final**

    Una vez completado el entrenamiento, se evalúa el modelo en el conjunto de prueba para medir su rendimiento real:

    - **Métricas de Rendimiento:** Dependiendo del tipo de problema, se utilizan métricas como precisión, **recall**, **F1-score**, o **error cuadrático medio para** evaluar la eficacia del modelo.

    - **Análisis de Resultados:** Se interpretan las métricas obtenidas para identificar posibles mejoras y entender las limitaciones del modelo.

10. **Implementación y Monitoreo**

    Tras la evaluación, el modelo se despliega en un entorno de producción donde se supervisa su rendimiento:

    - **Monitoreo Continuo:** Es esencial monitorear el rendimiento del modelo en datos reales para detectar posibles degradaciones debido a cambios en los datos subyacentes.

- **Actualizaciones y Reentrenamiento:** A medida que cambian los datos, puede ser necesario actualizar o reentrenar el modelo para mantener su relevancia y precisión.

2.1 Trabajos Anteriores

Anteriormente, se han realizado varios estudios relacionados con el análisis de datos en la Fórmula 1, enfocados en la predicción de resultados y optimización de estrategias de carrera. Algunos de estos trabajos, como los de **David Morán Montero** (2022) [29] y **Ana Urquidi Castro** (2022) [30], han utilizado técnicas avanzadas de Machine Learning para abordar estos desafíos.

1. **David Morán [29]** se centró en el uso de algoritmos de machine learning para agrupar circuitos y predecir los resultados de las carreras, basándose en características como el rendimiento de los pilotos y las condiciones del circuito. Tambien desarrolla una plataforma web que facilita la visualización interactiva y dinámica de los datos de telemetría y estadísticas generales de la Fórmula 1, permitiendo que cualquier aficionado, incluso sin experiencia previa en ingeniería de datos, pueda comprender y explorar esta información.

2. Por su parte, **Urquidi Castro [30]** aplicó mapas autoorganizados (SOM) para identificar patrones en el desempeño de los pilotos en diferentes circuitos, e identificar en qué circuitos destaca más un piloto y en cuáles tenderá a obtener peores posiciones, con el objetivo de que las escuderías analicen sus puntos fuertes y débiles para plantear mejor la estrategia de cara a futuras temporadas.

Estos estudios previos proporcionan una base sólida sobre la que construir y expandir el análisis de datos en la Fórmula 1. Al igual que estos trabajos, mi investigación busca avanzar en la aplicación de herramientas de aprendizaje automático para mejorar la precisión en la predicción de posiciones y la comprensión de las dinámicas de las carreras, abordando nuevas variables y enfoques para enriquecer el análisis de los datos de telemetría.

CÁPITULO 3:
OBJETIVOS

Este trabajo de fin de grado tiene como objetivo aplicar técnicas de machine learning para analizar y predecir el rendimiento en la Fórmula 1. Se aprovecharán datos de telemetría y otras fuentes para lograrlo. Los resultados obtenidos podrían ser útiles para los equipos de Fórmula 1, ayudándoles a optimizar sus estrategias y mejorar el rendimiento de sus vehículos, lo que tendría un impacto significativo en el ámbito deportivo. A partir de este objetivo general, los objetivos específicos son:

1. **Desarrollar modelos de machine learning para el análisis del rendimiento de pilotos y coches de Fórmula 1**. Se busca comprender los factores que afectan el rendimiento durante una carrera, como el tipo de los neumáticos, la velocidad en sectores clave, etc., así como identificar patrones únicos presentes en los datos de telemetría.

2. **Predecir la posición final de los pilotos en carrera**, teniendo en cuenta factores como la clasificación previa, estrategias de paradas en boxes y condiciones climáticas. Estos modelos tienen como fin proporcionar una visión predictiva del rendimiento de los pilotos, utilizando datos históricos y técnicas avanzadas de aprendizaje automático.

3. **Explorar el impacto de las estrategias de carrera y condiciones climáticas en el rendimiento de los pilotos**. Con el análisis de variables como los cambios de neumáticos y la meteorología, se pretende comprender cómo éstos afectan las posiciones finales en carrera y optimizar las decisiones durante la carrera.

4. **Optimizar los modelos de predicción desarrollados mediante metodologías iterativas e incrementales**. A lo largo del proyecto, se aplicarán diferentes enfoques de modelado (como MLP, LSTM y redes Transformer), además de técnicas de ajuste, como el balanceo de clases y el aumento del conjunto de datos, para mejorar la precisión de las predicciones. La mejora en la precisión se evaluará mediante métricas específicas, como el error medio absoluto (MAE) y la raíz del error cuadrático medio (RMSE).

5. **Convertir los datos de telemetría en información útil**, que pueda ayudar a optimizar la configuración vehicular y las estrategias de carrera, como la selección óptima de cuando cambiar los neumáticos según la telemetría. Esto implica analizar los datos recolectados en tiempo real y postcarrera para identificar oportunidades de mejora continua en el rendimiento de los pilotos y los coches.

6. **Evaluar la capacidad de generalización de los modelos predictivos desarrollados**. Probar su aplicabilidad en diferentes contextos, como temporadas de carreras distintas o escenarios no previstos, para garantizar la robustez de las soluciones propuestas y evitar el sobreajuste, asegurando que el modelo funcione bien con datos nuevos.

En definitiva, el proyecto pretende no solo crear modelos predictivos para el análisis del rendimiento en la Fórmula 1, sino también contribuir al avance de la aplicación de técnicas de machine learning en el contexto de la ingeniería de datos aplicada al deporte.

CÁPITULO 4: TECNOLOGÍA Y HERRAMIENTAS

Para llevar a cabo este proyecto, se ha empleado una sería de tecnologías y herramientas que han resultado esenciales en cada una de las etapas. Algunas de ellas facilitaron el proceso de programación y desarrollo de la solución, mientras que otras fueron necesarias para el análisis de resultados y para garantizar el correcto funcionamiento del sistema. En este apartado se explicará brevemente cuáles son estas herramientas y las razones de su elección. Cada una desempeña un papel importante y contribuye al desarrollo del proyecto de manera específica, por lo que resulta relevante conocerlas en mayor detalle.

4.1 Lenguajes de Programación

4.1.1 Python

Se ha utilizado únicamente Python para la extracción y análisis de datos relaciones con Fórmula 1, así como para la implementación de los modelos desarrollados en este trabajo de fin de grado.

Python [31] es un lenguaje de programación de alto nivel popular en análisis de datos e inteligencia artificial por su simplicidad y versatilidad. Su sintaxis clara facilita el desarrollo rápido, y sus bibliotecas como Pandas, NumPy y Scikit-Learn permiten manipular datos y crear modelos predictivos de forma eficiente, lo cual lo hace ideal para este proyecto.

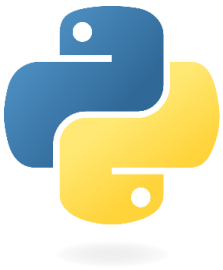

Ilustración 12: Logo oficial de Python [32]

Actualmente, Python es uno de los lenguajes más utilizados en análisis de datos, aprendizaje automático e IA. Su adopción generalizada en la industria se debe a su capacidad de adaptarse a una amplia variedad de aplicaciones, desde proyectos pequeños

hasta grandes sistemas en producción, consolidándose como una herramienta esencial para la ciencia de datos.

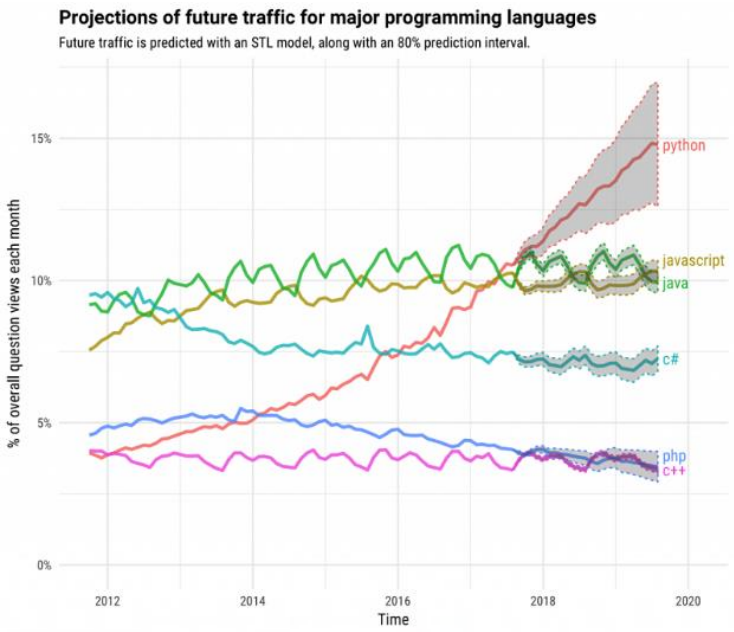

Ilustración 13: Comparación del uso de Python con respecto a otros lenguajes

En la ilustración 14 [33] se observa que **Python** ha tenido un crecimiento constante desde 2012 y se proyecta como el lenguaje más popular en el futuro.

La decisión de usar Python también fue debida a que los datos de Fórmula 1 se obtuvieron a través de una API que está implementada en Python. Para acceder a estos datos y extraerlos de forma efectiva, es necesario hacerlo utilizando este lenguaje, lo cual facilita la integración y el procesamiento de la información en el proyecto.

4.2 Entorno de Desarrollo

Para la extracción de los datos de Fórmula 1, se utilizó Visual Studio Code, mientras que para el análisis de los datos y el desarrollo de los modelos implementados se usó Google Colab.

4.2.1 Google Colab

Google Colab, o Google Colaboratory [34], es un entorno basado en Jupyter Notebook [35] que permite escribir y ejecutar código en Python directamente en el navegador, sin necesidad de configurar un entorno local. Es una plataforma gratuita proporcionada por

Google, muy popular en la comunidad de ciencia de datos y aprendizaje automático. Colab ofrece acceso gratuito a recursos de computación en la nube, como GPU y TPU, lo cual facilita el desarrollo y entrenamiento de modelos de machine learning de manera eficiente.

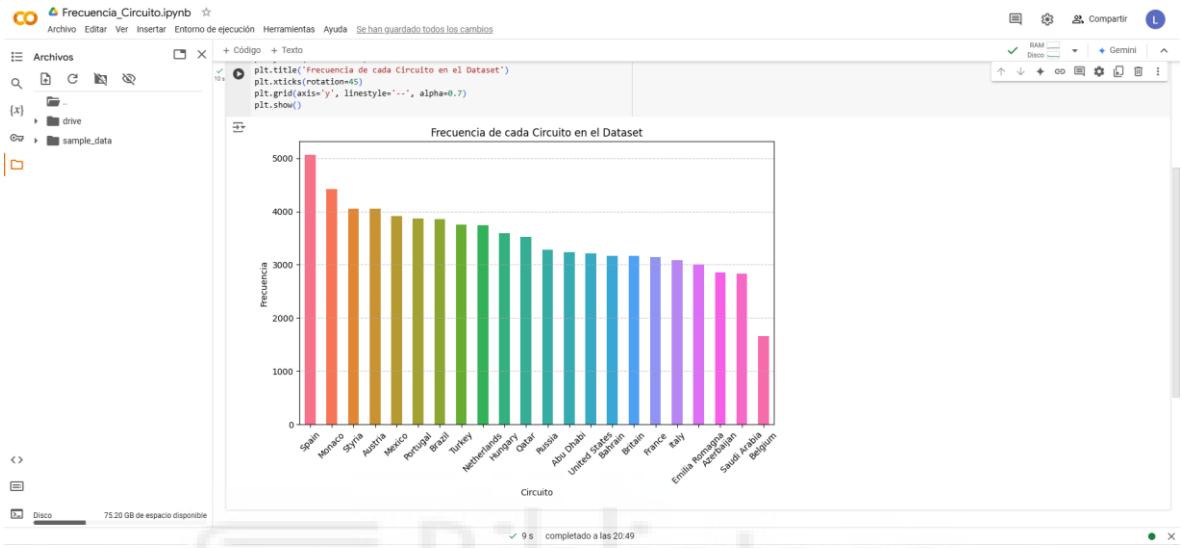

Ilustración 14: Ejemplo de la interfaz de usuario de Google Colab

Esta plataforma también permite colaborar en tiempo real, compartir cuadernos fácilmente, y trabajar con bibliotecas populares como TensorFlow, Keras, Pandas, entre otras, haciendo que sea accesible tanto para desarrolladores como para estudiantes.

4.2.2 Visual Studio Code

Debido a problemas de compatibilidad y conectividad al tratar de acceder a la API desde Google Colab, se decidió usar Visual Studio [36] Code para realizar la extracción de datos, ya que era una opción más adecuada para manejar las dependencias y el entorno local.

Visual Studio Code (VS Code) es un editor de código fuente gratuito y multiplataforma desarrollado por Microsoft. Está diseñado para ser ligero pero potente, ofreciendo soporte para múltiples lenguajes de programación como Python, JavaScript, C++, entre otros.

Es una herramienta muy popular para escribir, probar y depurar código, gracias a su interfaz amigable y la facilidad para instalar complementos que amplían sus funcionalidades.

Ilustración 15: Logo oficial de Visual Studio Code [37]

4.3 Tecnologías Externas

A continuación, se presentarán las tecnologías externas, herramientas, paquetes y bibliotecas empleadas en el proyecto.

4.3.1 FastF1

FastF1 [38] es una API construida en Python no oficial diseñada para facilitar el acceso y análisis de datos históricos y de telemetría de Fórmula 1. Esta construida sobre Pandas [39], y proporciona información detallada como tiempos de vuelta, telemetría de los vehículos, posiciones, datos meteorológicos y resultados de las sesiones. Es una herramienta ideal para quienes desean analizar el rendimiento en carreras y profundizar en los datos que ofrece cada evento.

4.3.2 Pandas

Pandas [39] es un paquete de código abierto desarrollado en Python utilizado para la manipulación y análisis de datos. Ofrece estructuras como DataFrames y Series que facilitan el manejo eficiente de grandes volúmenes de datos, permitiendo realizar operaciones como filtrado, agregación y transformación de forma sencilla.

Ilustración 16: Logo oficial de "Pandas" [40]

En este trabajo, Pandas se ha usado para cargar el dataset, convertir variables categóricas con One-Hot Encoding, dividir datos en X e Y, calcular métricas (MAE y MSE) por piloto y circuito, y crear tablas de resultados con predicciones y posiciones reales.

4.3.3 Matplotlib

Matplotlib [41] es una herramienta de visualización desarrollada en Python para crear gráficos y visualizaciones de datos. Permite generar una amplia variedad de gráficos, como líneas, barras y dispersión, de manera sencilla. Se utiliza principalmente para representar visualmente datos y resultados, facilitando la compresión y el análisis.

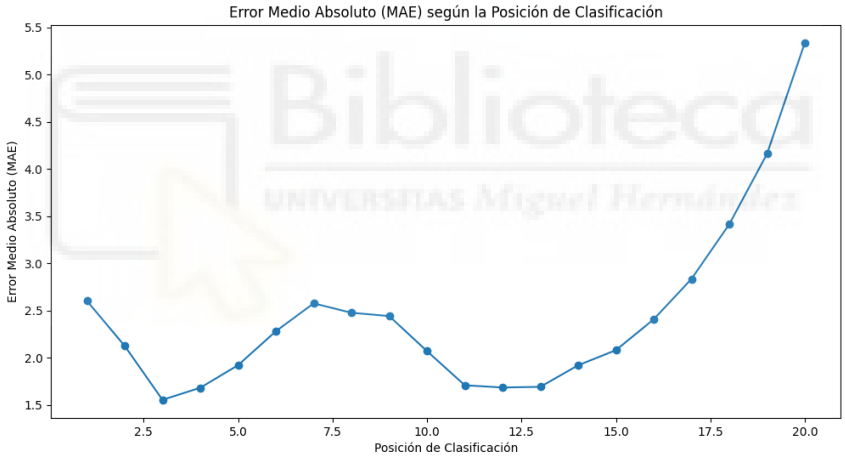

Ilustración 17: Ejemplo de visualización de la biblioteca Matplotlib

La gráfica de la ilustración 18 muestra un ejemplo del uso de esta biblioteca para visualizar cómo varía el **Error Medio Absoluto (MAE)** según la **posición de clasificación** de los pilotos.

4.3.4 NumPy

Numpy [42] es una biblioteca de Python que proporciona soporte para el cálculo numérico, especialmente con matrices y arreglos multidimensionales. Ofrece funciones matemáticas rápidas y eficientes, lo que la hace especialmente útil para el procesamiento de datos y el soporte a otras bibliotecas como Pandas y SciPy.

En este trabajo Numpy se utilizó principalmente para manipular datos eficientemente, realizar operaciones vectorizadas y copiar secuencias de vueltas sin afectar los datos originales.

4.3.5 PyTorch

Pytorch [43] es una biblioteca de aprendizaje automático desarrollada por Meta, utilizada principalmente para construir y entrenar redes neuronales. Es popular por su facilidad de uso y su naturaleza dinámica, lo que permite un desarrollo rápido y flexible de modelos de Deep learning. En esta biblioteca nos hemos basado para desarrollar las diferentes redes neuronales del proyecto.

Ilustración 18: Logo oficial de "Pytorch" [44]

4.3.6 FastAPI

FastAPI [45] es un framework web de Python que permite crear APIs de manera rápida y eficiente. Es conocido por su rendimiento, simplicidad y por ofrecer validación automática de datos. Se ha utilizado para obtener los datos necesario de Fórmula 1 mediante Python.

4.3.7 Scikit-Learn

Scikit-learn [46] es un paquete de código abierto desarrollado en Python diseñada para el aprendizaje automático y análisis predictivo. Proporciona herramientas para la clasificación, regresión, y agrupamiento, además de facilitar el preprocesamiento de datos

y la validación de modelos. Es ideal para construir y evaluar modelos de machine learning de manera rápida y eficiente.

Ilustración 19: Logo oficial de "Scikit-Learn" [47]

CÁPITULO 5: METODOLOGÍA

En este capítulo se presenta la metodología utilizada y la planificación general del proyecto, describiendo cómo se han organizado las tareas y los recursos para asegurar un desarrollo eficiente y estructurado. Se describen las etapas principales del proceso y cómo se han llevado a cabo las tareas clave para alcanzar los objetivos establecidos.

5.1 Metodología implementada

Para la realización de este trabajo se ha empleado una metodología basada en un enfoque **iterativo e incremental**, con características propias de un proceso de **desarrollo ágil**. El objetivo de esta metodología ha sido evolucionar los modelos de predicción de manera progresiva, mediante el análisis y la mejora constante de los resultados obtenidos en cada iteración. Esta metodología se caracteriza por:

- **Desarrollo Iterativo de Modelos**: Se plantearon y desarrollaron varios modelos, cada uno con un enfoque orientado a mejorar aspectos concretos del proceso de predicción. Cada modelo se construyó sobre la base del conocimiento adquirido en las etapas anteriores, permitiendo realizar ajustes y mejoras de manera continua.

- **Revisión y Mejora Continua**: Cada iteración de desarrollo se basó en los resultados y aprendizajes de los modelos previos. Esto permitió adaptar tanto los datos utilizados como la arquitectura del modelo en función de los resultados obtenidos y los desafíos encontrados. Esta forma de trabajo es característica del enfoque ágil, favoreciendo la adaptación rápida a los hallazgos durante el desarrollo.

- **Optimización y Experimentación**: Se llevaron a cabo diversas técnicas de ajuste para mejorar la precisión de los modelos, tales como el aumento del tamaño del conjunto de datos, técnicas de balanceo de clases, y la introducción de nuevos enfoques arquitectónicos (como redes LSTM o GRU). Esta fase fue clave para alcanzar un modelo robusto y confiable.

5.2 Planificación del trabajo

Para llevar a cabo este proyecto, se realizó una planificación general del trabajo, dividiéndolo en varias tareas específicas que aseguraran un avance progresivo y ordenado. La planificación se estructuró de la siguiente manera:

1. **Investigación y Ampliación de Conocimientos**: Estudio de redes neuronales, incluyendo conceptos teóricos y técnicas avanzadas, con el fin de obtener una base sólida para trabajar con modelos predictivos.

2. **Análisis y Recolección de Datos**: Revisión de las fuentes de datos y selección de las características más relevantes. Se recogen datos históricos de carreras de F1 para su análisis posterior.

3. **Definición del Problema y Enfoque del Proyecto**: Evaluación de los datos recolectados y planteamiento del problema a resolver, estableciendo los objetivos específicos del proyecto en función de los datos disponibles.

4. **Definición de Modelos y Metodología**: Planteamiento de los diferentes enfoques de modelado y elección de las técnicas más apropiadas. Se definen de las etapas de desarrollo y las herramientas a utilizar.

5. **Implementación de Modelos**: Desarrollo de los modelos de predicción, trabajando en etapas para implementar las versiones iniciales y luego ajustarlas y mejorarlas. Esta fase se dividió en la construcción de cada modelo y la implementación de diferentes técnicas de optimización y ajuste.

6. **Experimentación y Evaluación**: Realización de múltiples pruebas con cada modelo y análisis de resultados, con el fin de identificar mejoras y optimizar la precisión de las predicciones.

7. **Documentación**: Preparación de la documentación del proyecto, incluyendo la memoria técnica, la descripción detallada de los modelos desarrollados, y la elaboración de gráficos y resultados.

En la siguiente tabla se presenta un desglose detallado de la duración estimada en horas para la realización de cada tarea, así como las fechas de inicio y finalización correspondientes.

Tareas	**Tiempo estimado (h)**	**Inicio**	**Fin**
Investigación y Ampliación de Conocimientos	10	05/02/2024	16/02/2024
Análisis y Recolección de Datos	15	24/02/2024	28/02/2024
Definición del Problema y Enfoque del Proyecto	40	15/03/2024	19/04/2024
Definición de Modelos y Metodología	60	10/05/2024	23/06/2024
Implementación de Modelos	120	02/07/2024	11/11/2024
Experimentación y Evaluación	70	8/10/2024	24/11/2024
Documentación	50	01/10/2024	30/11/2024
N.º total de horas	365		

Tabla 1: Duración y periodo de cada tarea

5.3 Diagrama de Gantt

Para representar el avance del proyecto y las tareas realizadas en cada etapa, se ha utilizado un **diagrama de Gantt**. Este diagrama muestra la planificación temporal de las distintas actividades y su relación entre sí, permitiendo visualizar de manera clara las dependencias y los plazos establecidos para cada fase del proyecto.

El enfoque iterativo de la planificación permitió ajustar los tiempos y adaptarse a los imprevistos que surgieron durante el desarrollo, asegurando así una correcta gestión del proyecto y el cumplimiento de los objetivos establecidos.

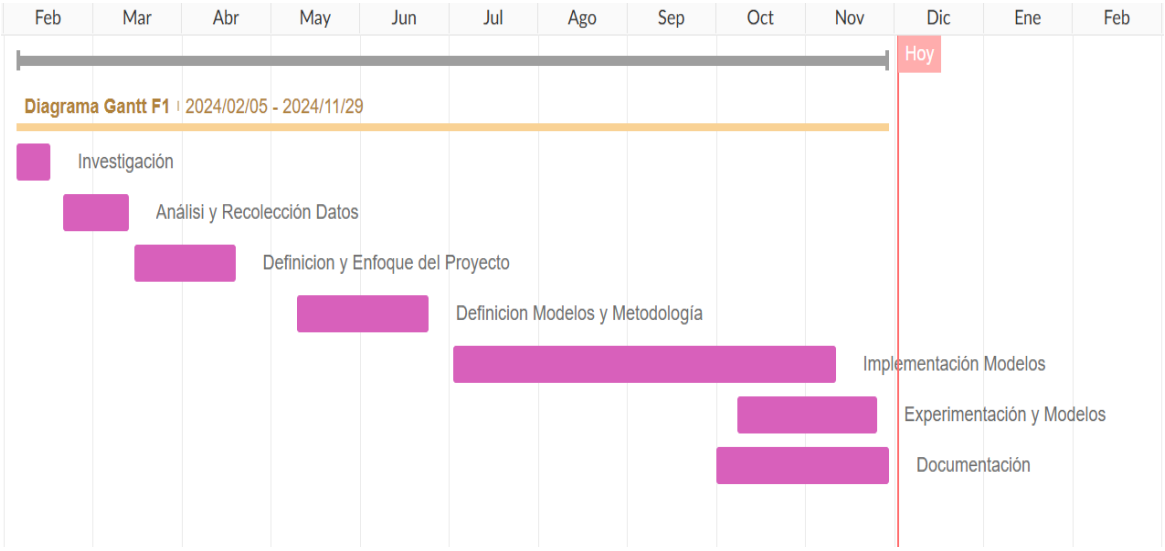

Ilustración 20: Diagrama de Gantt

La tarea 5 "**Implementación de Modelos"** se desarrolló entre julio y noviembre de 2024. Aunque inicialmente estaba planificada para llevarse a cabo de forma continua, hubo un parón de dos meses en agosto y septiembre, durante el cual el trabajo se detuvo por completo. Debido a esta interrupción, la duración total de la fase de experimentación se extendió más de lo previsto, lo que implicó retomar las actividades en octubre para poder completar las pruebas y ajustes necesarios antes de finalizar el plazo previsto. A pesar de la pausa, se logró reanudar el trabajo con éxito y avanzar en la optimización de los modelos.

CÁPITULO 6: DESARROLLO DEL PROYECTO Y RESULTADOS

En este capítulo se describe el conjunto de datos que se utilizó, especificando las columnas seleccionadas. También se explica el desarrollo del proyecto, abarcando en detalle cada uno de los modelos que se implementaron, asi como los problemas encontrados durante su desarrollo y cómo estos fueron enfrentados, reflejando los desafíos y soluciones aplicadas durante el proceso. Por último, se presentan los resultados obtenidos durante los análisis, evaluando el rendimiento de los modelos y discutiendo los hallazgos más importantes del proyecto.

6.1 Obtención y procesamiento de los datos

Para la extracción y el análisis de los datos, se decidió utilizar únicamente los datos de las sesiones de clasificación y de la carrera de las temporadas 2019-2023, excluyendo el año 2020. Esto se debe a que, debido a la pandemia de COVID-19, varias carreras fueron canceladas, lo cual generó inconsistencias en los datos de esa temporada de Fórmula 1.

La obtención y el procesamiento de los datos de Fórmula 1 se realizó a través de la biblioteca FastF1[46], que permitió acceder a datos detallados de telemetría, tiempos de vuelta, y otras estadísticas relevantes.

Mediante código en Python, se descargaron los datasets necesarios para el desarrollo de los distintos modelos de predicción. Estos datasets fueron enriquecidos añadiendo columnas y características adicionales según las necesidades de cada modelo, para mejorar la capacidad de análisis y predicción. Aunque se partió del mismo conjunto de datos básico, se realizaron modificaciones específicas para adaptarlo a los requerimientos de cada fase del proyecto.

A modo de ejemplo, la ilustración 21 muestra cómo se emplea la función *get_session()* de la biblioteca FastF1[38] para obtener los datos específicos de una sesión de carrera de Fórmula 1. Esta función permite acceder a información detallada de cada sesión, como la clasificación, prácticas o carreras, facilitando el análisis posterior de los datos.

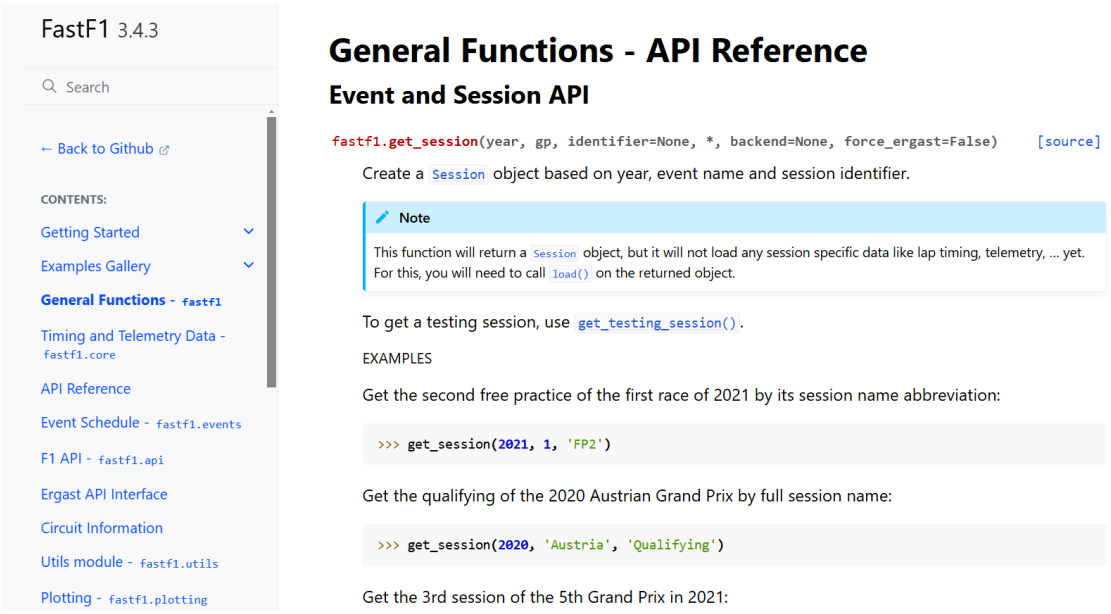

Ilustración 21: Ejemplo del uso de la API FastF1 para acceder a datos de sesiones de carreras de Fórmula 1

6.1.1 Descripción del dataset utilizado

Los datos contenidos en los datasets están organizados de manera que cada fila representa la información de una **vuelta individual realizada durante la carrera.** Tener la información organizada por vueltas nos ayuda a entender mejor cómo fue el rendimiento de cada piloto a lo largo de la carrera y a identificar patrones o cambios importantes entre una vuelta y otra.

Concretamente, se trabajó con **dos datasets** principales:

- **Dataset 1**: Este dataset contiene información general de cada carrera, incluyendo datos como tiempos de vuelta, número de vueltas, información sobre los sectores, y detalles del piloto y equipo. Este conjunto de datos fue la base inicial para el desarrollo de los modelos.

- **Dataset 2**: Este dataset se generó a partir del primero, añadiendo **características climáticas** adicionales, tales como la cantidad de lluvia, la temperatura del aire y la humedad. Estas nuevas variables se consideraron para analizar el impacto del clima en el rendimiento de los pilotos y optimizar las predicciones.

A continuación, se describen las columnas presentes en el primer dataset y su tipo de datos:

Característica	Descripción	Tipo
Driver	Identificador del piloto que participa en la carrera.	object
LapTime	Tiempo registrado por el piloto para completar una vuelta en el circuito.	object
LapNumber	Número de vuelta correspondiente al piloto en la carrera.	float64
Sector1Time, Sector2Time, Sector3Time	Tiempos registrados en los tres sectores que dividen cada vuelta	object
Stint	N.º de vueltas en los que un piloto continúa en la pista con el mismo juego de neumáticos	float64
PitInTime, PitOutTime	Momentos exactos en los que un piloto entra y sale de los pits durante una parada	object
SpeedI1, SpeedI2, SpeedFL, SpeedST	Velocidades registradas en diferentes puntos del circuito (intermedios y línea de meta)	float64
IsPersonalBest	Indica si la vuelta es la mejor vuelta personal de cada piloto	object
Compound	Tipo de neumático utilizado (blando, medio, duro)	object
TyreLife	Vida útil del neumático, es decir, cuántas vueltas lleva el compuesto utilizado	float64
FreshTyre	Neumático nuevo o no (usado)	bool
Team	El equipo de F1 al que pertenece el piloto	object
TrackStatus	Estado de la pista (pista despejada, bandera amarilla, etc.)	float64
Position	Posición en la que termine un piloto después de una carrera	float64
Deleted	Vuelta eliminada	bool
Circuit	Circuito en el que se celebró la carrera	object
Year	Año en el que se celebró la carrera	int64
SessionType	Tipo de sesión (Clasificación o Carrera)	object
Qualiposition	Posición en la que termina un piloto en la sesión de clasificación (posición en la parrilla al inicio de la carrera)	float64

Tabla 2: Descripción de las características del primer dataset

En la siguiente tabla se muestran únicamente las columnas del segundo dataset que no estuvieron presentes en el primero. Estas variables son parámetros meteorológicos que se miden durante las carreras de F1 para evaluar condiciones ambientales que pueden influir en el rendimiento y la seguridad.

Característica	**Descripción**	**Tipo**
AirTemp	Temperatura del aire en el circuito	float64
Humidity	Porcentaje de humedad del aire	float64
Pressure	Presión atmosférica	float64
Rainfall	Cantidad de lluvia caída, indicando si llueve durante la sesión (medida en mm)	object
TrackTemp	Temperatura de la pista	float64
WindDirecction	Dirección del viento	float64
WindSpeed	Velocidad del viento	float64

Tabla 3: Descripción de las características del segundo dataset

El uso de dos datasets permitió una mayor flexibilidad a la hora de diseñar los modelos, ya que se pudo realizar un análisis inicial con el conjunto de características básicas, y luego profundizar agregando información climática que podría influir en los resultados de las carreras.

6.2 Implementación y Resultados de los Modelos

En este apartado se presentan los modelos que hemos implementado y los resultados que hemos obtenido con cada uno de ellos. Cada modelo se enfoca en un aspecto concreto, como la predicción de la posición final de los pilotos según el circuito, las condiciones climáticas o el impacto de estrategias como la clasificación y el uso de neumáticos. A lo largo de este capítulo, analizamos estos resultados para valorar el desempeño de cada modelo y su capacidad para cumplir con los objetivos planteados.

6.2.1 MODELO 1: Predicción de la posición final de un piloto en función del circuito

El Modelo 1 tiene como objetivo predecir la posición final de un piloto en carreras de Fórmula 1 en función del circuito. Para lograr esto, se emplearon datos históricos que abarcan características del piloto, del circuito y de las condiciones de la carrera. Este modelo busca ofrecer una estimación razonable de las posiciones finales, destacando patrones y diferencias entre pilotos y circuitos.

En la siguiente tabla se muestra un breve resumen de las características de este modelo, las cuales se detallarán más adelante.

INPUT	ARQUITECTURA DEL MODELO	OUTPUT
Tiempos de vuelta (LapTime), tiempos de sectores (Sector1Time, Sector2Time, Sector3Time), tiempos acumulados de sesión (Sector1SessionTime, etc.), tiempos de boxes (PitInTime, PitOutTime), velocidades (SpeedI1, SpeedI2, SpeedFL, SpeedST), vida útil y estado de los neumáticos (TyreLife, IsPersonalBest), número de vuelta (LapNumber), y características categóricas (Driver, Team, Compound, Circuit)	MLP con 3 capas ocultas con 512, 256 y 128 neuronas	Posición final de un piloto en la carrera.

Tabla 4: Características del Modelo 1

6.2.1.1 DESARROLLO DEL MODELO

Para el **desarrollo del modelo**, se adoptó una metodología estructurada que incluyó varias etapas importantes:

- **Preprocesamiento:** Las columnas categóricas (como piloto, equipo y compuesto de neumáticos) fueron codificadas usando One-Hot Encoding. Los valores numéricos, como los tiempos de vuelta, se normalizaron para garantizar una escala uniforme. Además, los valores faltantes e infinitos se reemplazaron por ceros para evitar problemas durante el entrenamiento.

- **División de Datos:** Los datos se dividieron en un 80% para entrenamiento y un 20% para validación, asegurando una evaluación objetiva del modelo.

- **Entrenamiento:** El modelo se entrenó durante 100 épocas con la función de pérdida MSE y el optimizador Adam, ajustando los gradientes con recorte para evitar explosiones.

6.2.1.2 ARQUITECTURA DEL MODELO

En cuanto a la **arquitectura del modelo**, se utilizó una red neuronal profunda (MLP), configurada de la siguiente manera:

- **Capas Ocultas:** 3 capas ocultas (cada una con 512, 256 y 128 neuronas respectivamente) con activaciones ReLU para capturar patrones complejos.

- **Capa de Salida:** Diseñada para predecir la posición final de un piloto como un valor continuo.

- **Optimizador y Pérdida:** Se utilizó el optimizador Adam y la función de pérdida fue el Error Cuadrático Medio (MSE).

- **Hiperparámetros:** Learning rate de 0.0001, dropout del 50% y recorte de gradiente para evitar explosiones.

6.2.1.3 PROBLEMAS ENCONTRADOS

Durante el desarrollo, enfrentamos varios **desafíos**, algunos de ellos fueron los siguientes:

- **Valores Faltantes e Infinitos:** Algunos datos contenían valores problemáticos que afectaban el entrenamiento. Se solucionó reemplazando estos valores por ceros.
- **Nan en la Pérdida:** Esto ocurrió debido a problemas de estabilidad numérica. Se resolvió normalizando los datos y aplicando recorte de gradiente.
- **Tiempos Prolongados de Entrenamiento:** Debido a la complejidad del modelo y al volumen de datos, el entrenamiento era lento. Optamos por ajustar el batch size y trabajar inicialmente con subconjuntos de datos.

6.2.1.4 RESULTADOS DEL MODELO

Se han generado diversas visualizaciones para analizar el rendimiento del modelo.

POSICIONES REALES VS. PREDICHAS

En primer lugar, se han graficado las posiciones reales contra las predichas por el modelo para ver cómo se distribuyen los errores y si hay patrones evidentes.

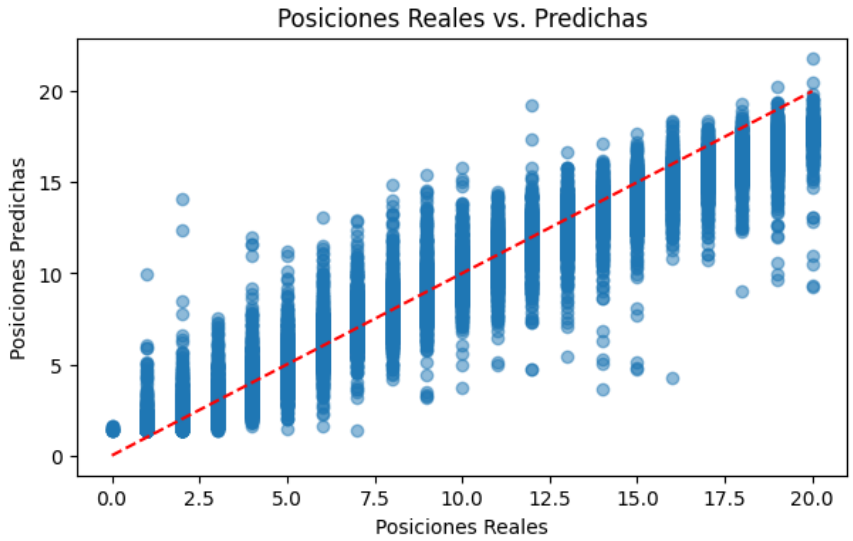

Ilustración 22: Gráfica de posiciones reales vs predichas

Este gráfico es un análisis básico de las **predicciones del modelo** comparadas con las **posiciones reales** de los pilotos en las carreras de F1. En el eje horizontal, tenemos las posiciones **reales** de los pilotos, que van del 1 al 20 (siendo 1 la mejor posición). En el eje vertical, se muestran las posiciones **predichas** por el modelo.

La **línea roja discontinua** muestra el comportamiento ideal del modelo. Si las posiciones predichas fueran exactamente iguales a las reales, todos los puntos caerían sobre esta línea. Cada uno de los **puntos azules** representa un piloto en una carrera específica, mostrando la diferencia entre su **posición real** y la **predicción** del modelo.

A partir de esta gráfica podemos concluir varias cosas:

1. Existe una **tendencia visible** que muestra que, a medida que la posición real aumenta, la diferencia entre la posición real y la predicha también lo hace. Esto puede indicar que el modelo tiene más dificultades para predecir correctamente las posiciones más altas (más cerca del primer lugar) y tiene mayor precisión en las posiciones más bajas (más cerca del 20).
2. La **dispersión de los puntos** alrededor de la línea roja sugiere que el modelo tiene errores de predicción, pero no de manera uniforme. Los errores parecen ser más grandes en algunas posiciones y más pequeños en otras, lo que puede reflejar la influencia de factores complejos, como las condiciones del circuito o la estrategia del piloto, que no están completamente capturados por el modelo.

DISTRIBUCIÓN DE PREDICCIONES Y VALORES REALES POR CIRCUITO

Tambien se ha hecho un análisis específico por circuito, mostrando cómo el modelo se comporta en diferentes condiciones, ya que las características del circuito pueden influir en el rendimiento de los pilotos.

El siguiente gráfico muestra la **distribución de las predicciones** versus las **posiciones reales** por circuito, es similar al anterior, pero ahora se incluye la distinción por **circuito**. Cada punto sigue representando una predicción del modelo, pero los puntos están coloreados según el circuito en el que ocurrió la carrera.

- **Eje X (Posiciones Reales):** Representa la posición real del piloto, desde la mejor (1) hasta la peor (20).
- **Eje Y (Posiciones Predichas):** Muestra la posición que el modelo predijo para cada piloto.
- **Línea roja discontinua:** Al igual que en el gráfico anterior, esta línea indica el comportamiento ideal. Si las predicciones fueran perfectas, todos los puntos deberían caer sobre esta línea.

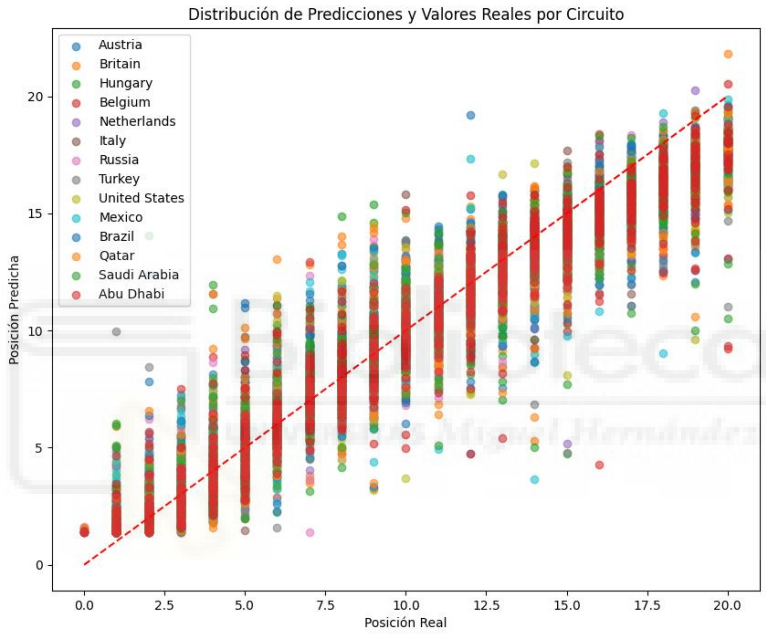

Ilustración 23: Gráfica de Distribución de Predicciones y Valores Reales por Circuito

Como podemos observar en la gráfica, los puntos están coloreados según el circuito, lo que nos permite ver si el modelo tiene un desempeño diferente en distintos circuitos. Esto también indica si el modelo tiene más dificultades para predecir posiciones de ciertos circuitos, o bien, que los circuitos tienen características que afectan más a la predicción.

Tambien se aprecia como a medida que las posiciones reales aumentan (de la posición 1 a la 20), tambien lo hacen las posiciones predichas. Esto indica que el modelo tiene cierta **tendencia** a predecir más errores en las posiciones más altas (cerca del primer lugar) y que las predicciones son más cercanas a las reales en las posiciones más bajas.

Además, algunas categorías de circuito parecen tener más **dispersión** en las predicciones, lo que podría significar que el modelo no está tan afinado para esos circuitos específicos. Este es el caso de **Hungría** (amarillo) y **Qatar** (celeste), en ellos vemos como hay puntos alejados de la línea ideal, lo que indica predicciones menos precisas. Tambien es el caso de **Brasil** (verde oscuro) y **México** (azul), donde se observa una mayor dispersión en posiciones intermedias (10-15). Esto podría deberse a factores no capturados completamente por las características actuales, como las condiciones específicas del circuito o la estrategia de neumáticos en ese circuito.

Del mismo modo, también se ven posiciones bien predichas. En algunos circuitos como **Austria** (azul claro), **Abu Dhabi** (rojo oscuro) e **Italy** (**Monza**) (gris) el modelo parece tener una mejor precisión, con los puntos mucho más cerca de la línea discontinua roja. Esto sugiere que, en esos circuitos, las características del piloto y del circuito están más alineadas con el modelo.

En resumen, este gráfico ayuda a mostrar si el modelo tiene un sesgo hacia ciertos circuitos y permite identificar qué circuitos pueden necesitar ajustes adicionales para mejorar la precisión del modelo.

ERROR CUADRÁTICO MEDIO (MSE)

Asi mismo, se ha calculado el **error cuadrático medio (MSE) para cada piloto y circuito**, lo cual me ha permitido observar qué pilotos tienen un mayor error de predicción y si existen circuitos donde el modelo funciona mejor o peor. El **MSE** se calcula como la media de los cuadrados de las diferencias entre las posiciones reales y las predicciones del modelo. Cuanto mayor sea el **MSE**, mayor es el error en la predicción para ese piloto y circuito.

A continuación, se procede a explicar algunas de las gráficas del **MSE** obtenidas, para ello se han seleccionado aquellas que muestran tendencias relevantes o casos interesantes de comportamiento del modelo. Esta selección se realiza para evitar la repetición innecesaria y centrarse en los ejemplos más representativos y analíticamente valiosos.

Es necesario saber previamente que en la gráfica:

- **Eje X (Piloto):** Los pilotos se muestran en el eje horizontal
- **Eje Y (MSE):** En el eje vertical, se muestra el valor del **MSE** para cada piloto en ese circuito.

Este gráfico es útil para ver cómo el modelo se comporta con respecto a cada piloto en diferentes circuitos.

En primer lugar, nos encontramos con el **circuito** de **Austria**. La gráfica para este circuito destaca por tener errores bajos y consistentes, y por lo tanto una **buena precisión**. Muestra un MSE bastante bajo y homogéneo entre los pilotos. No hay picos significativos ni variaciones extremas, lo que sugiere que el modelo realiza predicciones consistentes y precisas en este circuito.

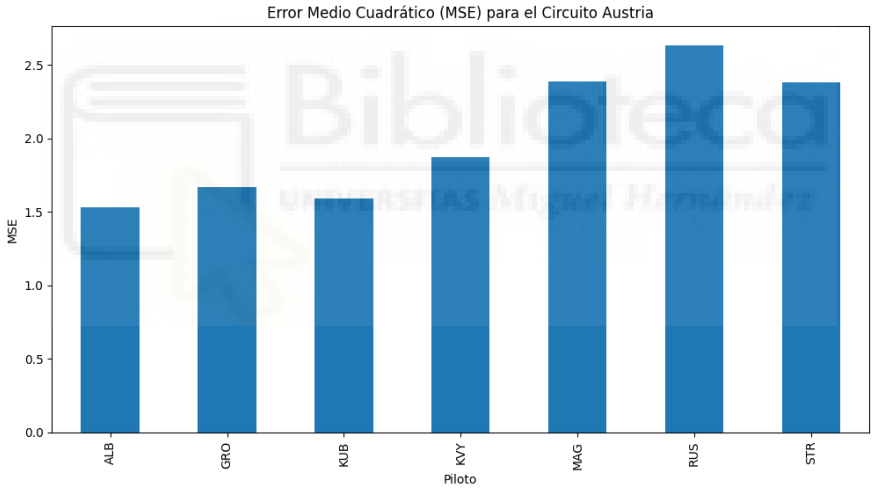

Ilustración 24: MSE para el Circuito de Austria

Se puede concluir que el modelo muestra un buen desempeño en el circuito de Austria, lo que sugiere que las características seleccionadas para entrenar la red MLP permiten capturar patrones que generalizan bien en este caso. Esto indica que el modelo puede identificar relaciones relevantes entre las características y las posiciones predichas en este circuito, reflejando un ajuste adecuado para estas condiciones específicas.

En segundo lugar, tenemos el **circuito** de **Hungría**, donde se observa una **dispersión mucho mayor** en el **MSE**, ya que vemos en pilotos como Grosjean con un error extremadamente alto, que indica que el modelo no logra capturar patrones relevantes para este caso.

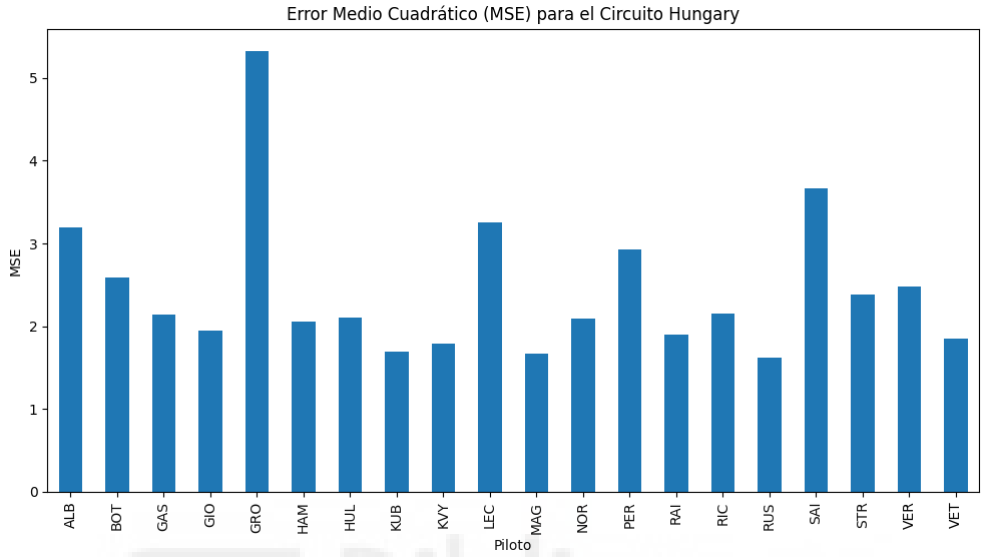

Ilustración 25: MSE para el Circuito de Hungría

Mientras tanto, otros pilotos tienen errores más moderados. Este pico significativo destaca un fallo evidente del modelo en este caso.

Por lo tanto, podemos concluir que la alta dispersión y el error elevado en Grosjean sugieren que las características seleccionadas para el modelo no logran capturar patrones relevantes que expliquen el rendimiento de este piloto en este circuito, Esto podría deberse a datos insuficientes, valores atípicos o una relación más compleja entre las variables y el resultado que no ha sido bien aprendida.

En tercer lugar, tenemos el **circuito** de **México**, la gráfica para este circuito es muy relevante porque muestra un **error desproporcionado en Norris (outlier extremo)**, siendo un ejemplo claro de un caso particular en el que el modelo falla de forma drástica

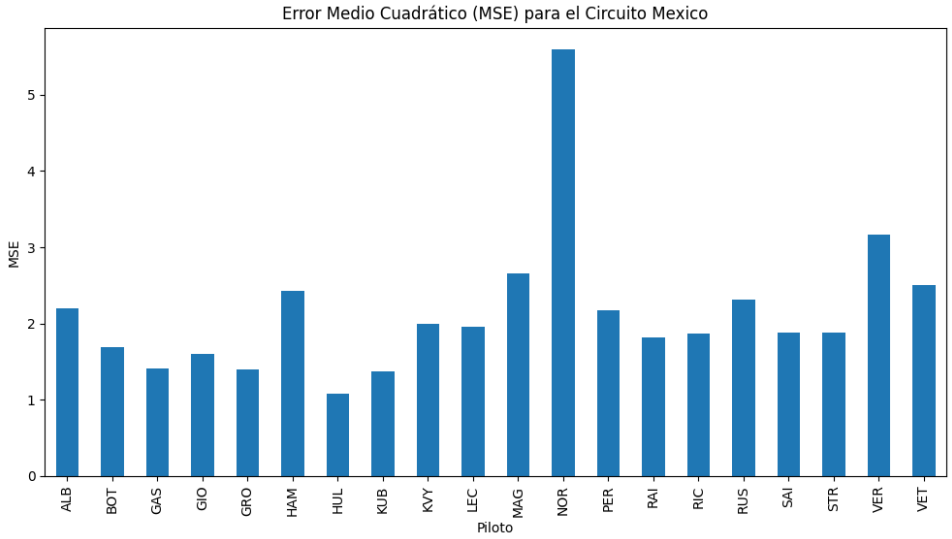

Ilustración 26: MSE para el Circuito de México

En contraste, los errores para otros pilotos son más consistentes y dentro de un rango razonable. Este caso resalta cómo el modelo puede fallar drásticamente en situaciones específicas, posiblemente debido a datos insuficientes o características que no reflejan adecuadamente el rendimiento de Norris. Esto evidencia la importancia de identificar y tratar outliers en los datos para reducir su impacto en la calidad global del modelo.

Por último, pasamos al **circuito** de **Arabia Saudí**. Aquí vemos un error inusual en un piloto, pues se muestra un pico claro en Hulkenberg, un error significativamente más alto que el resto, mientras que el **MSE** para otros pilotos se mantiene dentro de un rango aceptable.

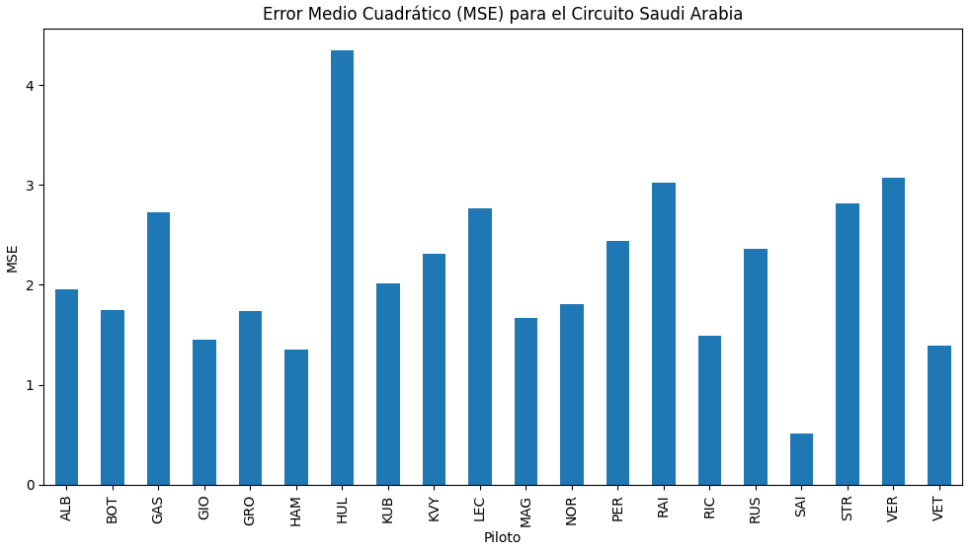

Ilustración 27: MSE para el circuito de Arabia Saudí

De nuevo, esto es debido a características del piloto o de la carrera que no están bien representadas en los datos. Identificar y tratar estos casos ayudaría a mejorar la capacidad del modelo para generalizar.

COMPARACIÓN DE PREDICCIONES Y VALORES REALES PARA CADA PILOTO

Para finalizar con este modelo, se ha generado tambien una visualización para cada circuito donde se compara las posiciones finales reales predichas por piloto, lo que da una visión más detallada sobre qué pilotos el modelo predice con mayor precisión.

Las gráficas mostradas a continuación reflejan claramente que los resultados obtenidos no fueron satisfactorios, ya que el modelo tiende a predecir posiciones finales cercanas al 20 para la mayoría de los pilotos, mostrando una falta de precisión significativa. Esto se debe a varios factores: La **selección de características** pudo haber sido uno de los puntos débiles del modelo, ya que las variables utilizadas no lograron capturar las dinámicas complejas de las carreras, lo que dificultó que el modelo pudiera diferenciar correctamente el rendimiento de los pilotos. Además, es probable que el **diseño de la arquitectura** del modelo o su **implementación técnica** no estuvieran bien ajustados para este problema. Aspectos como el número de neuronas, capas o incluso las funciones de activación podrían no haber sido los más adecuados, lo que afectó directamente la capacidad del modelo para aprender.

Aparte de los problemas en la selección de características y la arquitectura, el modelo tampoco logró capturar adecuadamente las **posiciones reales de cada piloto**, una variable que formaba parte de las entradas del modelo. Esto indica que, aunque esta información estaba presente en los datos, el modelo no pudo establecer relaciones claras entre esta variable y las posiciones finales que intentaba predecir. Por último, al ser un modelo inicial y en una **fase más experimental**, no se llevaron a cabo pruebas exhaustivas, lo que limitó aún más su desempeño.

Se analizan las gráficas seleccionadas que ilustran estas limitaciones, donde las comparaciones de las posiciones reales se muestran en azul, y las posiciones predichas en verde. Además, el **eje X** (eje horizontal) representa los **pilotos**, y cada barra representa la

comparación entre su posición real y la predicha por el modelo, y el **eje Y** (eje vertical) representa la **posición final** del piloto en la carrera.

En primer lugar, tenemos el **circuito** de **México**, donde vemos errores generales con un patrón.

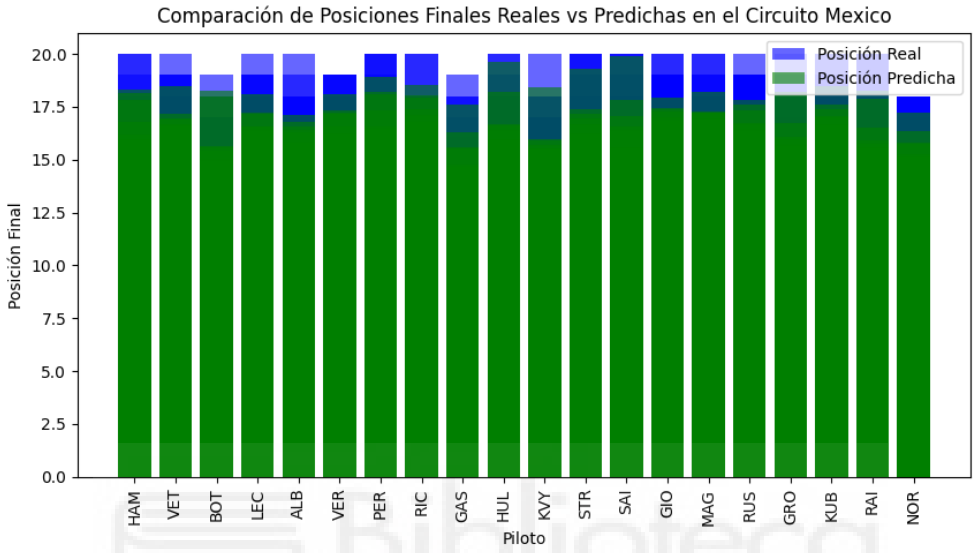

Ilustración 28: comparación de posiciones en el Circuito de México

Aquí podemos ver como las posiciones predichas como he mencionado anteriormente están mayormente cerca de la posición 20, pero lo interesante aquí es que el patrón se repite para casi todos los pilotos, reflejando que el modelo tenía una tendencia fuerte a otorgar posiciones bajas.

En segundo lugar, tenemos el **circuito** de **Hungría**:

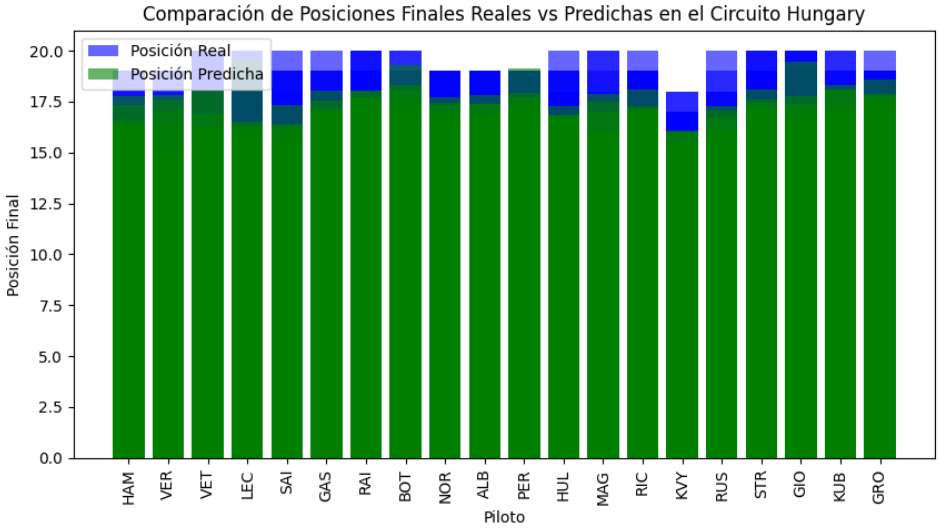

Ilustración 29: Comparación de posiciones finales en el circuito de Hungría

Aunque tambien hay muchas predicciones cercanas al 20, esta gráfica muestra una mayor variabilidad en los resultados predichos para diferentes pilotos. Esto indica que el modelo intentaba capturar diferencias, aunque sin éxito significativo.

Por último, el **circuito** de **Estados Unidos** nos muestra diferencias marcadas en algunos pilotos.

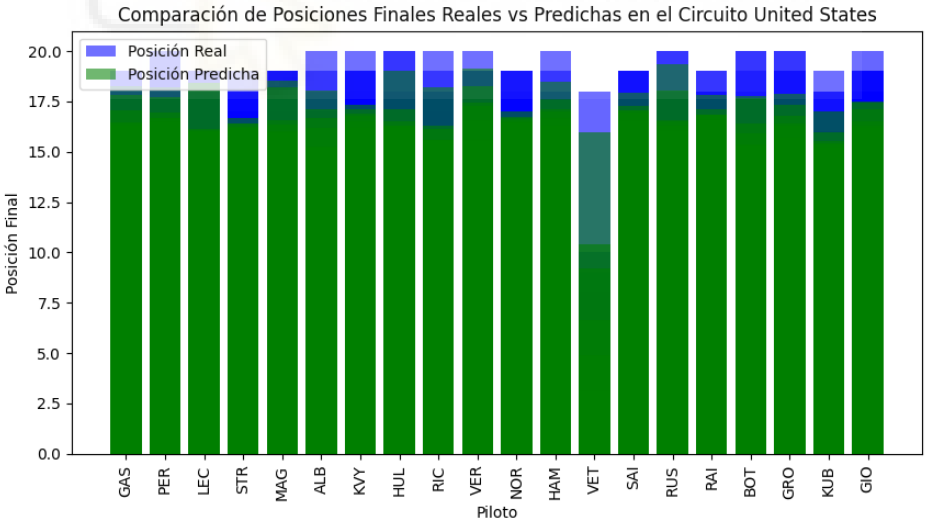

Ilustración 30: Comparación de posiciones en el circuito de EE. UU.

Aunque los resultados siguen siendo en su mayoría cercanos al 20, hay pilotos como Vettel donde la diferencia entre la posición real y predicha es más marcada. Esto destaca como un caso particular de error.

Esta gráfica muestra que el modelo no solo tendía a predecir mal para la mayoría, sino que también cometía errores más evidentes en ciertos pilotos.

6.2.1.5 CONCLUSIÓN DEL MODELO

En conclusión, el Modelo 1 no logró alcanzar los resultados esperados en cuanto a precisión, pero resultó clave para identificar las limitaciones y las áreas que necesitaban mejoras en futuros desarrollos. Fue una base importante para experimentar y tomar decisiones más fundamentadas en la construcción de los modelos siguientes, los cuales, gracias a los aprendizajes obtenidos, lograron resultados mucho más acertados y cercanos a la realdad. Este primer intento no debe verse como un fallo, sino como un paso necesario dentro del proceso de aprendizaje, optimización y mejora continua.

6.2.2 MODELO 2: Predicción de la posición final del piloto en base a las condiciones climáticas.

El objetivo de este modelo fue predecir las posiciones finales de los pilotos, considerando las condiciones climáticas durante las carreras. Para ello, se utilizó una red neuronal MLP, al igual que en el primer modelo, ya que este modelo está construido a partir del anterior. Se integraron variables relacionadas con el clima, como la temperatura del aire, la humedad, la presión atmosférica, la dirección y velocidad del viento, y la temperatura de la pista. No obstante, durante el desarrollo del modelo, nos encontramos con un desafío importante: los datos climáticos disponibles resultaron ser insuficientes, ya que una gran parte de los campos del dataset estaban vacíos. A pesar de esto, se intentaron dos versiones del modelo con el fin de evaluar su capacidad para predecir las posiciones en función de diversas combinaciones de factores climáticos, con especial énfasis en cómo estos influyen en el rendimiento de los pilotos. Sin embargo, ambos enfoques resultaron ser fallidos y los resultados obtenidos no fueron válidos.

A continuación, se presenta una tabla resumen de las características y la arquitectura del modelo, que incluye tanto las variables de entrada como la estructura utilizada en ambas versiones.

Esta tabla refleja los aspectos comunes de las dos versiones y la diferencia en sus configuraciones:

INPUT	ARQUITECTURA DEL MODELO	OUTPUT
LapTime, LapNumber, PitOutTime, PitInTime, Sector1Time, Sector2Time, Sector3Time, Speedl1, Speedl2, SpeedFL, SpeedST, Stint, TrackStatus, TyreLife, FreshTyre, IsPersonalBest, Position, Compound, Driver, Team, Circuito, Year, AirTemp (Temperatura del aire), Humidity (Humedad), Pressure (Presión atmosférica), Rainfall (Pista mojada), TrackTemp (Temperatura de la pista), WindDirection (Dirección del viento), WindSpeed (Velocidad del viento)	**Versión 1:** MLP con 3 capas ocultas con 256, 256 y 128 neuronas. Utiliza Batch Normalization, Dropout y activación ReLU para evitar sobreajuste. Optimización con Adam (tasa de aprendizaje 0.001). **Versión 2:** MLP con 256, 256 y 128 neuronas. Uso de Batch Normalization, Dropout y activación ReLU. Optimización con Adam (tasa de aprendizaje 0.0005).	Posición final de un piloto en la carrera.

Tabla 5: Características del Modelo 2 (Versión 1 y 2)

6.2.2.1 PROBLEMAS ENCONTRADOS

Como mencioné anteriormente, durante el desarrollo de este modelo surgieron varios desafíos que impidieron obtener los resultados deseados. En este apartado se detallan los problemas principales que se presentaron durante el proceso de implementación y que afectaron el rendimiento del modelo.

1. **Insuficiencia de datos climáticos:** Los datos climáticos disponibles resultaron ser insuficientes para poder hacer una predicción precisa. La mayoría de los campos relacionados con las condiciones climáticas estaban vacíos en el dataset,

lo que afecto a la capacidad del modelo para aprender correctamente las relaciones entre las variables climáticas y las posiciones de los pilotos.

2. **Datos incompletos y nulos:** A pesar de los esfuerzos por limpiar los datos, los valores faltantes en algunas variables clave redujeron la calidad de los datos de entrada, y el manejo de estos valores nulos no fue suficiente para mejorar el rendimiento del modelo en ambas versiones.

3. **Resultados no válidos:** Debido a los problemas con los datos, los modelos no produjeron los resultados esperados. A pesar de los esfuerzos por ajustar la arquitectura y los parámetros de las redes neuronales, el modelo no mostraba resultados coherentes, lo que me llevó a dar por fallido el Modelo 2.

4. **Problemas con el preprocesamiento:** También se realizó una limpieza exhaustiva de los datos y se implementaron técnicas como la normalización y codificación de las características, pero los resultados seguían sin ser satisfactorios.

6.2.2.2 RESULTADOS DEL MODELO 2 – VERSIÓN 1

En esta sección se presentan los resultados del modelo para su primera versión, los cuales han sido evaluados bajo diferentes condiciones climáticas para analizar la precisión de las predicciones de las posiciones finales de los pilotos. A través de las gráficas que siguen, se pretendía comparar las posiciones reales con las predichas, considerando variables como la temperatura del aire, la humedad, y la dirección y velocidad del viento. Aunque se esperaba que el modelo lograra un mejor ajuste, los resultados obtenidos indican una notable diferencia entre las predicciones y los valores reales. A continuación, se explica con más detalle cada uno de los agrupamientos y sus implicaciones.

DISPERSIÓN ENTRE LA POSICIÓN REAL VS PREDICHA

Esta gráfica muestra la comparación entre las posiciones reales y las predicciones del modelo para cada carrera. En ella, cada punto representa una predicción, con el eje X indicando la posición real de los pilotos y el eje Y mostrando la posición que predijo el modelo.

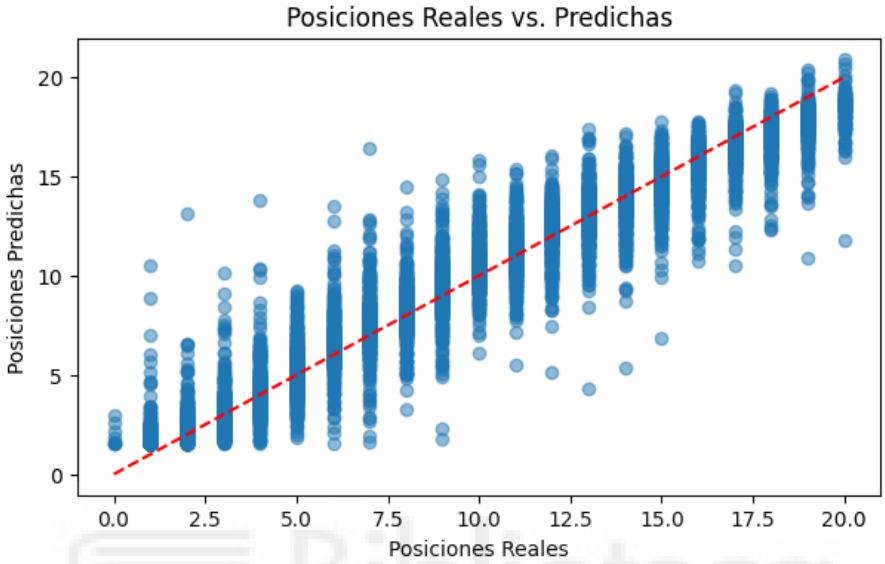

Ilustración 31: Comparación entre las posiciones reales y predichas por el Modelo 2

La línea roja discontinua es la línea de regresión, que nos muestra cómo deberían alinearse las dos variables. Cuanto más cerca estén los puntos de esta línea, más precisas serán las predicciones. Sin embargo, en este caso, se puede ver que las predicciones se alejan bastante de la línea, lo que afirma que el modelo tiene dificultades para predecir con exactitud las posiciones de los pilotos.

ANÁLISIS DE POSICIONES PREDICHAS EN BASE A LAS CONDICIONES CLIMÁTICAS COMBINADAS

Para este análisis se ha generado la siguiente gráfica que muestra cómo las predicciones del modelo varían según diferentes combinaciones de condiciones climáticas. Los puntos están coloreados de acuerdo con las condiciones combinadas (AirTemp_Humidity, WindSpeed_WindDirection, etc.).

Se puede observar que, a pesar de las variaciones en las condiciones climáticas, la distribución de las posiciones predichas no parece tener una correlación clara con estas

combinaciones, lo que confirma una vez más que el modelo no ha logrado captar correctamente la influencia del clima en las posiciones de los pilotos.

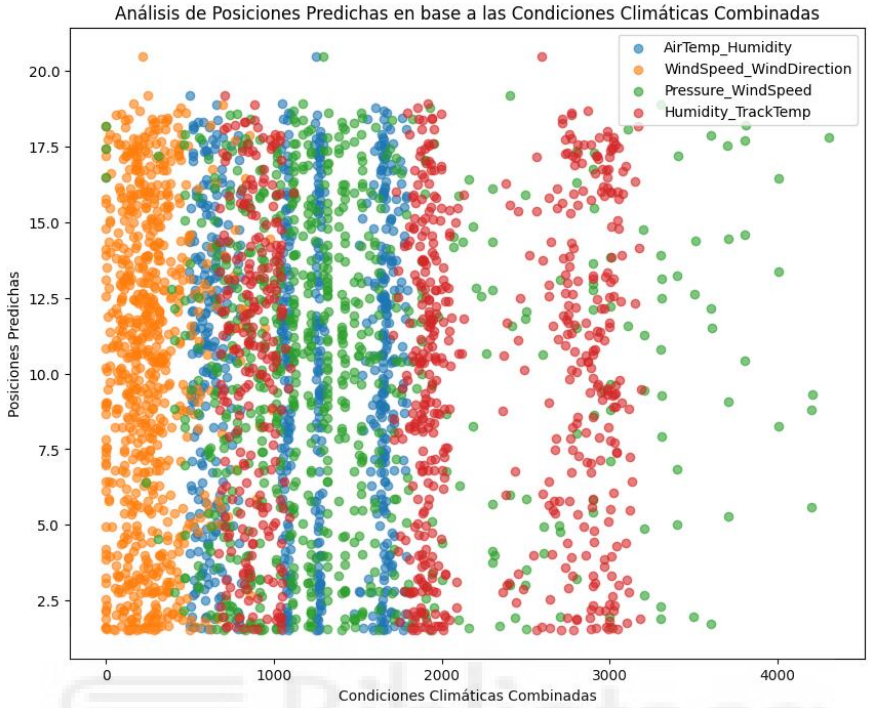

Ilustración 32: Impacto de las Condiciones Climáticas Combinadas en las Posiciones Predichas

Este tipo de análisis es útil para identificar si hay variables que el modelo podría estar ignorando o si su capacidad para generalizar bajo diferentes condiciones no está optimizada. La distribución dispersa de los puntos también indica que las predicciones del modelo varían de manera impredecible, lo que reforzó la idea de que el modelo podría necesitar ajustes adicionales o más características para mejorar su rendimiento.

DISTRIBUCIÓN DE PERDICCIONES Y VALORES REALES EN FUNCIÓN DE CADA COMBINACIÓN

En este estudio se han generado gráficos para analizar cómo se distribuyen las posiciones reales y las predicciones del modelo cuando se agrupan según los rangos de cada combinación de factores climáticos vista antes. Para evitar redundancia, aquí explicare únicamente uno de estos gráfico, el gráfico correspondiente a la combinación **AirTemp_Humidity**.

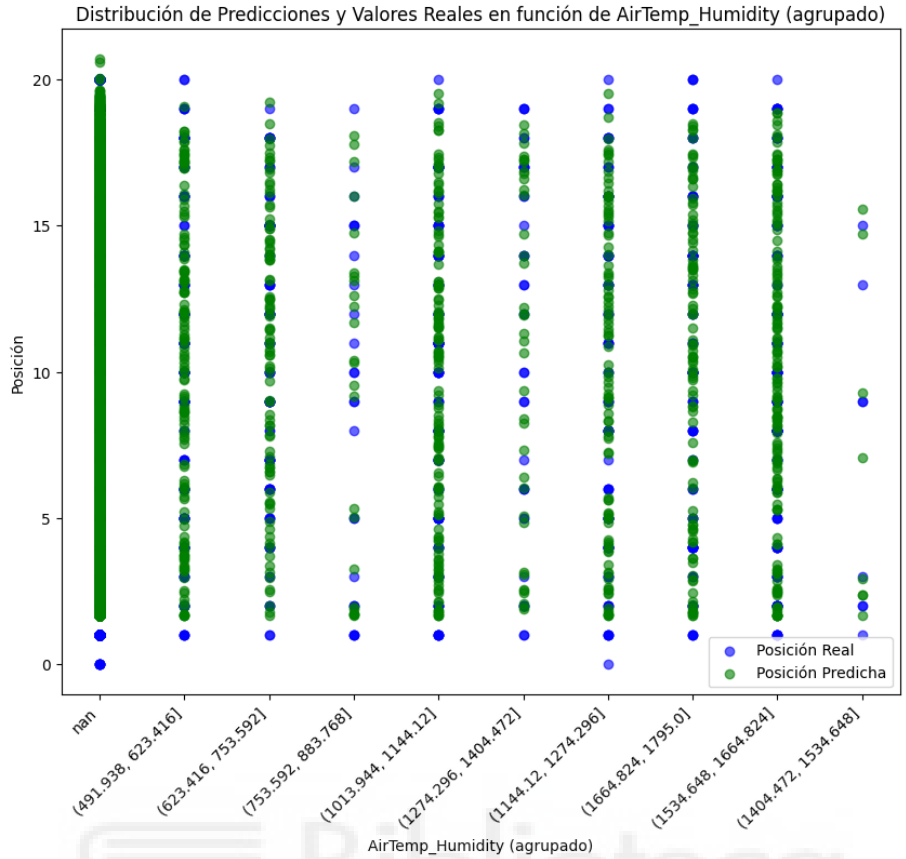

Ilustración 33: Distribución de Posiciones Reales y Predichas en función de la Combinación de AirTemp_Humidity

El gráfico muestra la relación entre las posiciones reales y las predicciones del modelo, con los datos agrupados según la combinación de temperatura del aire y humedad. En el eje X se representan los rangos agrupados de AirTemp_Humidity, mientras que el eje Y muestra las posiciones finales de los pilotos. Los puntos azules corresponden a las posiciones reales y los puntos verdes a las predicciones del modelo. A pesar de las variaciones en las condiciones climáticas, la dispersión de los puntos indica que el modelo no ha sido capaz de capturar correctamente la influencia del clima sobre las posiciones de los pilotos. Esto indica que las predicciones no siguen un patrón claro, lo que refleja las limitaciones del modelo.

COMPARACIÓN DE POSOICIONES REALES VS PREDICHAS BAJO CADA COMBINACIÓN CLIMÁTICA

En este análisis he generado gráficas para todas las combinaciones climáticas y segmentadas en rangos específicos de valores para cada una de las condiciones climáticas involucradas. Esto permite observar cómo varían las predicciones en función de las

diferentes condiciones climáticas dentro de rangos determinados, lo que ofrece una perspectiva más detallada de la influencia de cada factor climático, pero con el fin de evitar repeticiones innecesarias y hacer la explicación más concisa, solo se procederá a explicar una gráfica por combinación climática.

Combinación AirTemp_Humidity: nan

En esta gráfica, se muestra la comparación entre las posiciones reales y las predichas por el modelo para cada piloto, agrupadas según las condiciones de 'AirTemp_Humidity'.

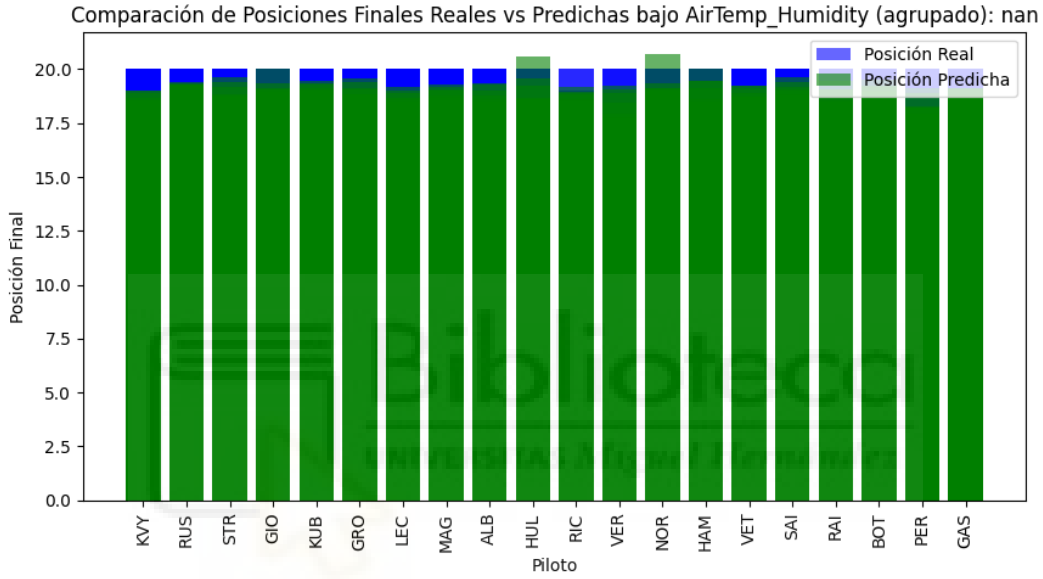

Ilustración 34: Comparación de Posiciones Finales Reales vs Predicciones bajo la variable AirTemp_Humidity

Aunque las barras están alineadas, lo que podría indicar una aparente correspondencia entre las posiciones reales y las predichas, esto no refleja correctamente el comportamiento real de las posiciones de los pilotos, ya que el modelo no está captando correctamente las posiciones reales, y es que, en la todos los casos, todas las posiciones reales están muy cercanas a la peor posición (20), y en la vida real, los pilotos no ocupan todos valores tan altos, lo que demuestra que el modelo está teniendo dificultades para predecir correctamente las posiciones en estas condiciones climáticas.

Además, el valor "**nan**" en el eje X indica que hay datos faltantes o no disponibles para ciertas combinaciones específicas de 'AirTemp_Humidity'. Esto es debido a la falta de datos para esas condiciones como se ha explicado anteriormente. Los valores "nan" reflejan una deficiencia en los datos utilizados, lo que afecta negativamente la precisión del modelo en esas combinaciones específicas de condiciones climáticas.

Combinación WindSpeed_WindDirection: [-1.01, 101.05]

Aquí se muestra la comparación de las posiciones finales reales y predichas de los pilotos Pierre Gasly (**GAS**), Sergio Pérez (**PER**), Charles Leclerc (**LEC**), Lewis Hamilton (**HAM**), Sebastian Vettel (**VET**), y Max Verstappen (**VER),** bajo la combinación climática "WindSpeed_WindDirection" con un rango de valores entre -1.01 y 101.05.

Como se observa, la mayoría de los pilotos tienen una posición final real cercana a la más alta (alrededor de 17-18), lo cual, de nuevo, no refleja correctamente el rendimiento de estos pilotos en la realidad.

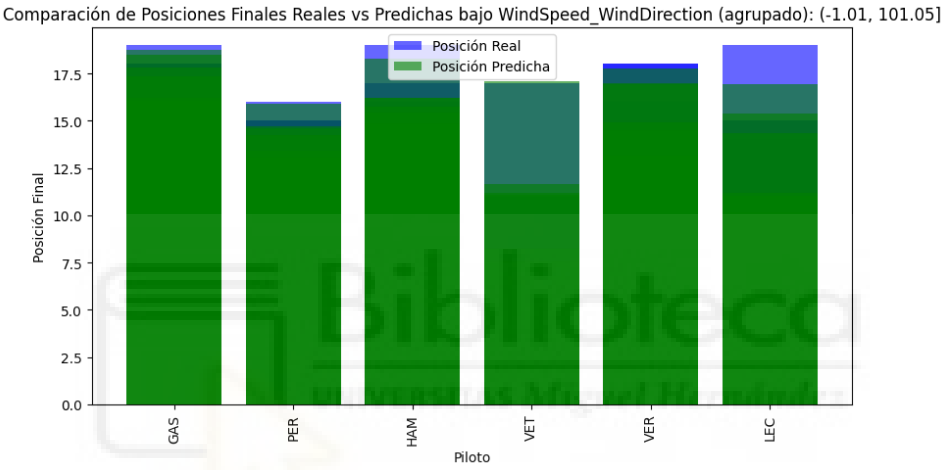

Ilustración 35: Comparación de Posiciones Finales Reales vs Predicciones bajo la variable WindSpeed_WindDirection

Al igual que en las otras gráficas, las posiciones predichas no coinciden de manera precisa con las reales, ya que las barras verdes (predicciones) no se alinean bien con las barras azules (reales). A pesar de que las predicciones para algunos pilotos, como Gasly o Hamilton, parecen estar más cerca de las posiciones reales, todavía hay una diferencia significativa en las predicciones.

Además, el modelo parece estar dando resultados consistentes para ciertos pilotos (como Verstappen y Leclerc), pero no de forma precisa, ya que las diferencias en las barras también son notables.

Combinación Pressure_WindSpeed: [2152.58, 2583.096]

Esta gráfica muestra una comparación entre las posiciones finales reales y las predichas por el modelo para algunos pilotos basándose en el análisis bajo las condiciones combinadas de presión y velocidad del viento (agrupadas en el rango de [**2152.58, 2583.096**]). A diferencia de las gráficas previas, esta parece mostrar resultados más realistas, ya que, por ejemplo, Verstappen, quien es conocido por obtener las mejores posiciones en la vida real, tiene valores muy bajos en la predicción. Esto es un indicio de que el modelo ha logrado captar de manera más precisa las dinámicas del rendimiento de los pilotos en relación con las condiciones climáticas asociadas a la presión y la velocidad del viento.

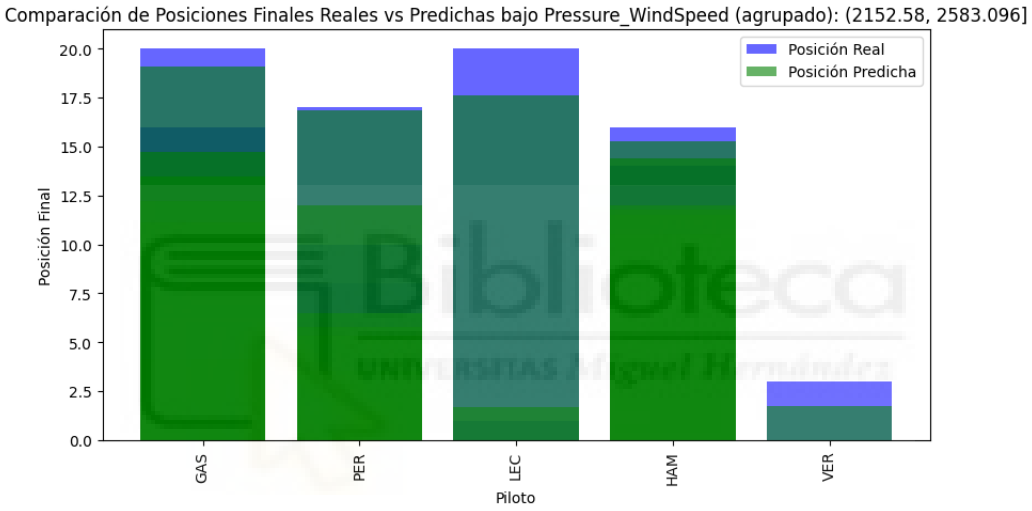

Ilustración 36: Comparación de Posiciones Finales Reales vs Predicciones bajo la variable Pressure_WindSpeed

Sin embargo, aunque el modelo ha mostrado una mejor alineación con la realidad para algunos pilotos, como Verstappen, se siguen observando algunas discrepancias para otros. Esto indica que, aunque el modelo ha mejorado en esta categoría, aún es necesario realizar ajustes adicionales para optimizar su capacidad de predicción para todos los pilotos y condiciones posibles.

Combinación Humidity_TrackTemp: [1927.875., 2179.5]

Por último, aquí vemos la comparación entre las posiciones reales y las posiciones predichas para los pilotos bajo las condiciones de humedad y temperatura de la pista (agrupadas en el rango de Humidity_TrackTemp: (1927.875, 2179.5)).

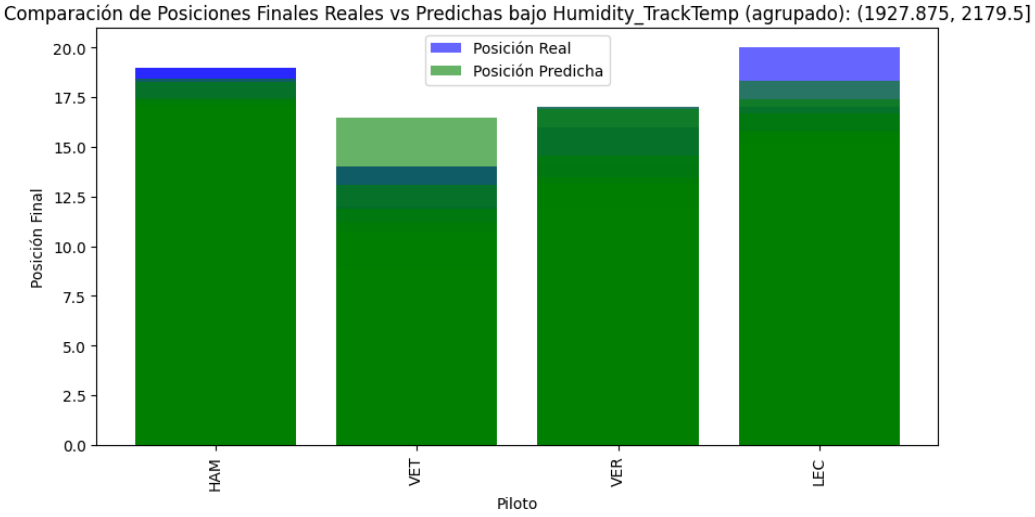

Ilustración 37: Comparación de Posiciones Finales Reales vs Predicciones bajo la variable Humidity_TrackTemp

En primer lugar, se puede observar que las posiciones predichas para los pilotos como Hamilton (HAM) y Verstappen (VER) están alineadas con las posiciones reales, pero siguen siendo bastante altas en comparación con lo que se esperaría de estos pilotos en la vida real, quienes suelen estar en las primeras posiciones. Esto indica que el modelo no está capturando de manera efectiva las dinámicas de rendimiento de los pilotos más competitivos bajo estas condiciones.

Además, para pilotos como Leclerc (LEC) y Vettel (VET), las posiciones predichas están desviadas, con predicciones que colocan a los pilotos en posiciones mucho más altas de lo que sería realista.

6.2.2.3 RESULTADOS DEL MODELO 2 – VERSIÓN 2

Como mencioné anteriormente, en un intento por mejorar los resultados, se implementó una segunda versión del modelo. Sin embargo, a pesar de los esfuerzos por afinar los parámetros y las características, no se logró una mejora significativa en las predicciones.

ANÁLISIS DE PREDICCIONES EN FUNCIÓN DE LAS CONDICIONES CLIMÁTICAS

El primer análisis que se obtuvo generó los resultados que se muestran a continuación en las gráficas, las cuales reflejan las comparaciones entre las posiciones reales y las predichas en función de una **combinación climática** concreta (se han generado gráficas para todas las combinaciones, pero solo se explicarán algunas de las para evitar redundancias).

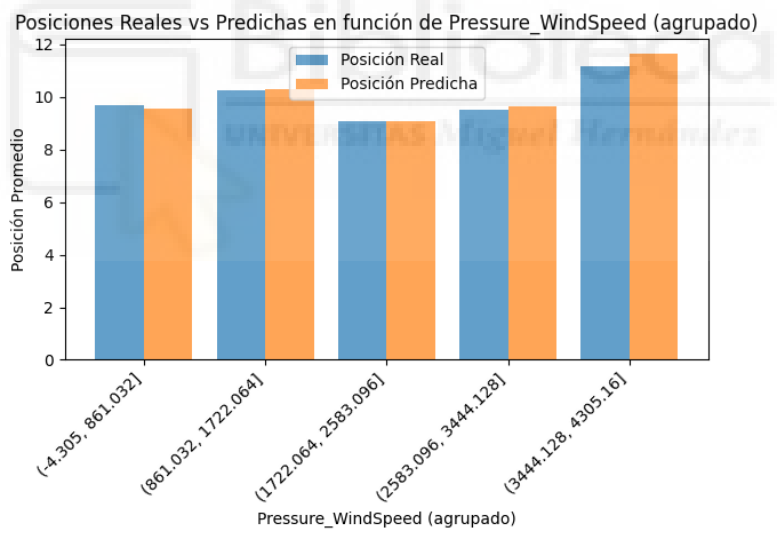

Ilustración 38: Comparación de Posiciones Reales vs Predichas en función de Pressure_WindSpeed

Se puede observar que el modelo tiende a predecir posiciones que se agrupan principalmente entre las posiciones 10 y 12, sin cubrir un rango más amplio de posiciones. Esta limitación podrías deberse a que el modelo no está capturando adecuadamente la variabilidad necesaria para predecir posiciones extremas.

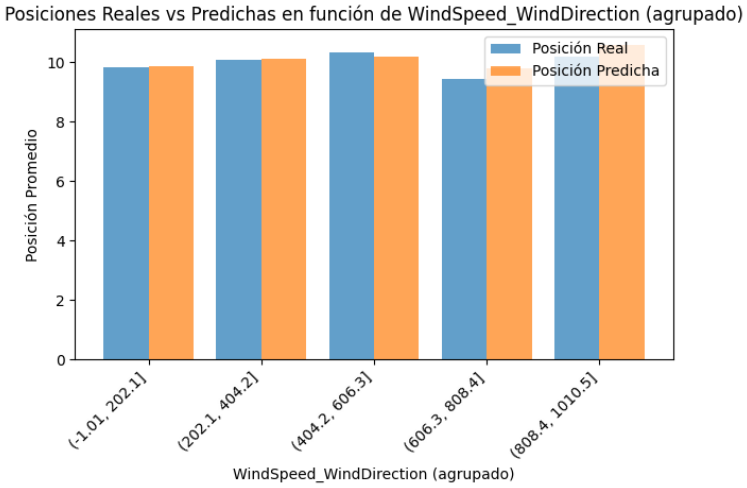

Ilustración 39: Comparación de Posiciones Reales vs Predichas en función de WindSpeed_WindDirection

Este comportamiento podría ser el resultado de un ajuste incorrecto de los parámetros del modelo o de una insuficiencia en los datos de entrenamiento, que no reflejan adecuadamente todos los posibles escenarios de las posiciones de los pilotos. Además, las predicciones de los pilotos más alejados al podio parecen ser menos precisas.

Los rangos en el eje X de las gráficas representan las agrupaciones de las variables climáticas utilizadas para generar las predicciones. Cada bin o intervalo en el eje X muestra un rango específico de valores para cada factor climático (por ejemplo, AirTemp, WindSpeed, etc.). Por ejemplo, un rango como "(-1.01, 202.1)" indica que el modelo ha agrupado todas las observaciones que tienen valores dentro de este intervalo para una determinada variable climática. Este agrupamiento es útil para reducir la complejidad del análisis y facilitar la visualización de los resultados, pero también puede limitar la capacidad del modelo para hacer predicciones precisas en intervalos específicos que no estén bien representados en los datos de entrenamiento.

ERROR MEDIO CUADRÁTICO (MSE) EN FUNCIÓN DE CADA COMBINACIÓN CLIMÁTICA

Los gráficos que se muestran reflejan el error medio cuadrático (MSE) calculado para distintas combinaciones climáticas, lo que proporciona una visión de la precisión del modelo según las condiciones atmosféricas. Estos resultados sugieren lo siguiente:

1. **Para AirTemp_Humidity**: El MSE varía entre los diferentes rangos de la combinación de temperatura y humedad del aire, pero no muestra grandes diferencias. Los valores más bajos de MSE se encuentran en los rangos intermedios, como el rango de (753.592, 1013.944), lo que indica que el modelo podría estar funcionando mejor en estas condiciones, pero sin una mejora destacada.

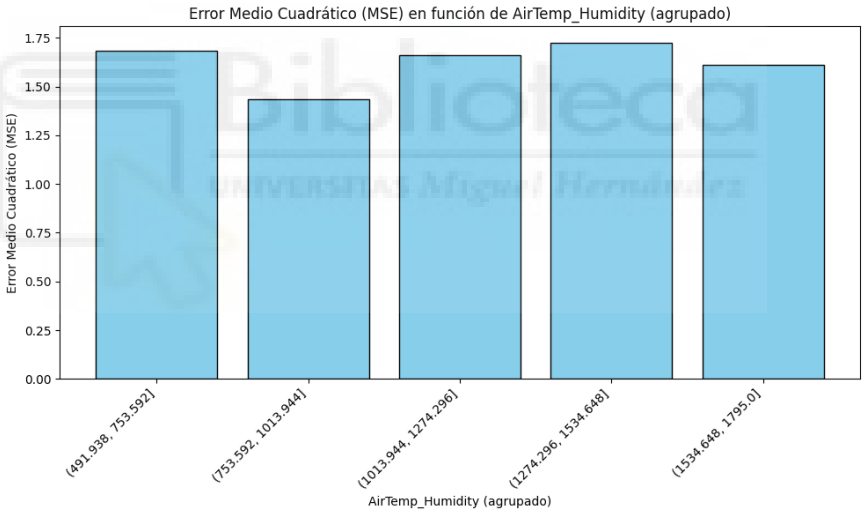

Ilustración 40: MSE en función de la combinación climática AirTemo_Humidity

2. **Para WindSpeed_WindDirection**: El error aumenta considerablemente para los rangos de vientos más fuertes, como en el intervalo (606.3, 808.4), lo que sugiere que el modelo tiene más dificultades para predecir las posiciones bajo estas condiciones. Este aumento en el MSE indica que el modelo no está capturando adecuadamente los efectos de las direcciones del viento y las velocidades elevadas en el desempeño de los pilotos.

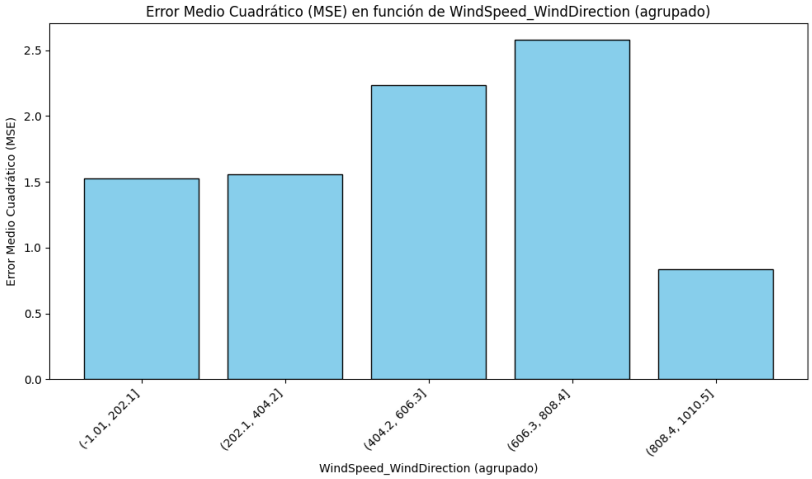

Ilustración 41:MSE en función de la combinación climática WindSpeed_WindDirection

3. **Para Pressure_WindSpeed**: Similar a los resultados anteriores, los valores del MSE muestran una mayor variabilidad en los intervalos más extremos, como el rango (3444.128, 4305.16), donde el error es mayor. Esto es porque el modelo también tiene dificultades para manejar condiciones con altas variaciones en la presión y velocidad del viento.

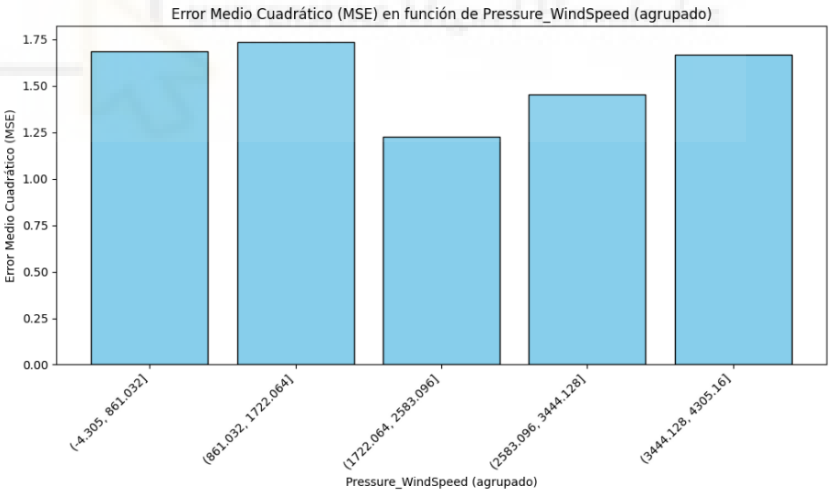

Ilustración 42: MSE en función de la combinación climática Pressure_WindSpeed

En resumen, los resultados sugieren que el modelo tiende a ser menos preciso bajo ciertas combinaciones extremas de condiciones climáticas, especialmente con **vientos más fuertes y presiones inusuales**. Aunque no se observa una mejora significativa con respecto al primer análisis, estos gráficos ayudan a entender mejor qué factores climáticos afectan la precisión de las predicciones.

6.2.3 MODELO 3: Evaluación de la influencia de la posición de clasificación en la posición final en carrera

Este modelo se desarrolló con el objetivo de predecir las posiciones finales de los pilotos de Fórmula 1 en carrera basándose principalmente en sus **posiciones de clasificación** y otras variables relacionadas, como tiempos de vuelta, sectores, velocidad y condiciones de los neumáticos. Durante el proceso, se construyó un modelo de red neuronal multicapa (MLP) que fue entrenado y evaluado utilizando datos históricos de múltiples temporadas. El objetivo principal fue analizar cómo la posición de clasificación influye en el resultado final y qué factores adicionales podrían aportar al rendimiento en carrera.

En la siguiente tabla podemos ver un breve resumen de las características de este modelo:

INPUT	ARQUITECTURA DEL MODELO	OUTPUT
Posición de clasificación (QualiPosition), tiempos de vuelta (LapTime), tiempos de sectores (Sector1Time, Sector2Time, Sector3Time), tiempos acumulados de sesión (Sector1SessionTime, Sector2SessionTime, Sector3SessionTime), velocidades (SpeedI1, SpeedI2, SpeedFL, SpeedST), vida útil y estado de los neumáticos (TyreLife, FreshTyre, IsPersonalBest), características categóricas como el piloto (Driver), el equipo (Team), el compuesto del neumático (Compound) y el circuito (Circuit).	MLP con 3 capas ocultas con 64, 32 y 1 neuronas. Activación ReLU y optimización con Adam	Posición final de un piloto en la carrera.

Tabla 6: Características del Modelo 3

Este modelo representa una mejora directa del Modelo 1, por lo que su desarrollo mantuvo una estructura similar, pero incorporó elementos adicionales que enriquecieron el análisis. El preprocesamiento incluyó las mismas técnicas utilizadas anteriormente, como la codificación One-Hot para variables categóricas y la normalización de características numéricas, pero se añadieron nuevas variables relacionadas con el rendimiento en carrera,

como el estado de los neumáticos. Estas incorporaciones ampliaron la cantidad de información disponible para el modelo y, en consecuencia, su capacidad predictiva.

La arquitectura del modelo también mantuvo la estructura de tres capas ocultas, pero se adaptó el número de neuronas para manejar las nuevas características de entrada. Al igual que en el modelo anterior, se utilizaron la función de pérdida MSE y el optimizador Adam, manteniendo prácticas como el recorte de gradientes para asegurar la estabilidad del entrenamiento. Esta evolución permitió que el modelo ofreciera predicciones más detalladas y precisas, facilitando un análisis más profundo del rendimiento de pilotos.

6.2.3.1 PROBLEMAS ENCONTRADOS

Durante el desarrollo del modelo, surgieron varios problemas que se tuvieron que resolver para que todo funcionara correctamente. Estos desafíos ayudaron a mejorar el modelo y ajustar tanto los datos como el proceso de entrenamiento. A continuación, se describen los principales problemas que encontramos y cómo se abordaron:

- **Problemas con los datos**: Algunas columnas presentaban valores faltantes o inconsistencias, como tiempos registrados como cadenas en lugar de valores numéricos. Esto requirió convertir formatos, imputar valores faltantes y eliminar valores infinitos para garantizar la integridad de los datos.

- **Incremento de dimensionalidad**: La integración de nuevas características, como tiempos de vuelta, velocidades y estado de los neumáticos, aumentó significativamente la dimensionalidad, provocando inicialmente que el modelo devolviera valores NaN. Esto se solucionó ajustando la normalización y escalado de los datos.

- **Errores técnicos**: Hubo problemas relacionados con índices en arrays y estructuras de datos que causaron fallos durante el entrenamiento y la evaluación. Estos se resolvieron revisando el flujo de datos y ajustando el código correspondiente.

- **Ajustes durante el entrenamiento**: La aparición de valores NaN en las primeras iteraciones del modelo hizo necesario revisar las técnicas de preprocesamiento y el manejo de características para garantizar estabilidad durante el entrenamiento.

A pesar de su complejidad, cada uno de estos problemas se resolvió de forma efectiva, contribuyendo a un modelo más preciso y robusto.

6.2.3.2 RESULTADOS DEL MODELO 3

Los resultados obtenidos en este modelo se presentan a través de diversas gráficas y métricas que permiten analizar su desempeño. Se han generado visualizaciones como diagramas de dispersión que comparan las posiciones reales con las predichas, gráficos de caja para observar la distribución de las predicciones por posición de clasificación y barras que muestran errores medios y probabilidades de mantener una buena posición en carrera. Estas representaciones gráficas ofrecen una visión clara de la precisión del modelo y destacan tanto sus fortalezas como las áreas en las que puede mejorar

DISTRIBUCION DE POSICIONES FINALES PREDICHAS POR POSICION DE CLASIFICACIÓN

A continuación, se muestra una gráfica de caja que muestra la distribución de las posiciones finales predichas por el modelo para cada posición de clasificación inicial. En el eje horizontal se representan las posiciones de clasificación (del 1 al 20), mientras que en el eje vertical se reflejan las posiciones finales predichas en la carrera. Cada caja ilustra la mediana, el rango intercuartílico (entre el primer y tercer cuartil) y los valores atípicos de las predicciones para cada posición de clasificación.

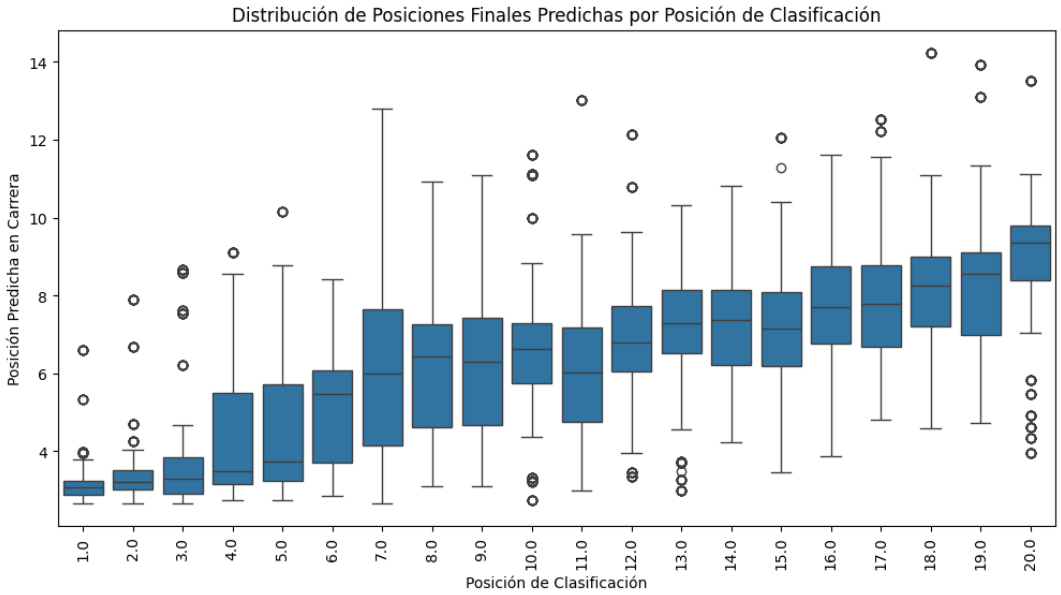

Ilustración 43: Distribución de Predicciones de Posiciones Finales Según la Posición de Clasificación Inicial

El objetivo de esta gráfica es evaluar la dispersión y la tendencia central de las predicciones según la posición inicial en la clasificación. Las cajas más compactas reflejan una menor variabilidad en las predicciones, mientras que las más extendidas indican una mayor dispersión.

Se observa que los pilotos que clasifican en posiciones más altas (cercanas a 1) tienden a tener predicciones de posiciones finales también altas, con una variabilidad reducida. Sin embargo, a medida que la posición de clasificación empeora (hacia 20), la variabilidad en las posiciones finales predichas aumenta significativamente. Esto sugiere que el modelo tiene mayor dificultad para predecir con precisión las posiciones finales de pilotos que comienzan en posiciones más bajas.

DESVIACIÓN MEDIA DE LA POSICIÓN REAL VS PREDICHA

La siguiente gráfica muestra la **desviación media** entre las posiciones reales finales y las posiciones predichas por el modelo, agrupadas según la posición inicial en la clasificación. Indica cómo de lejos, en promedio, están las predicciones del modelo respecto a las posiciones reales finales para cada posición de clasificación.

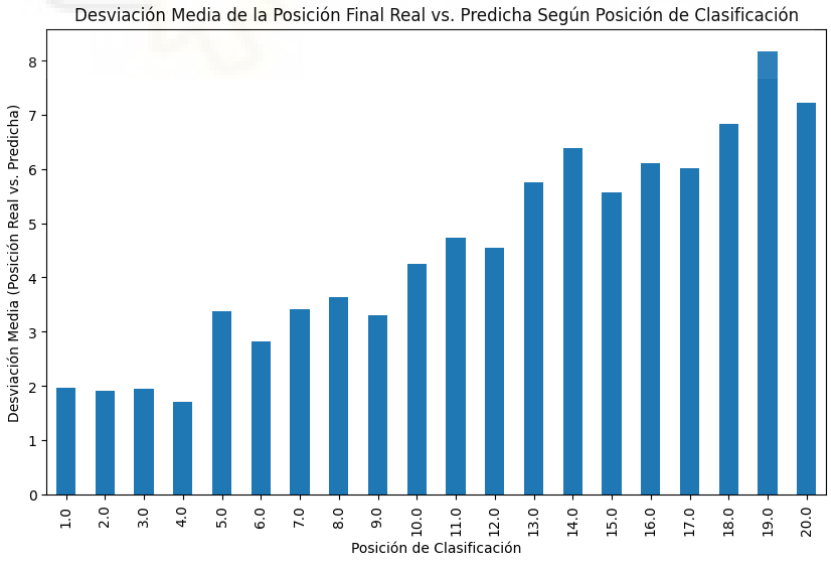

Ilustración 44: Relación entre la Desviación Media de Posiciones Predichas y la Posición de Clasificación

Este análisis refuerza las conclusiones obtenidas en la gráfica de dispersión anterior. Ambas muestran que el modelo es más preciso y consistente al predecir las posiciones finales de los pilotos que parten desde posiciones de clasificación altas, ya que presentan

menor variabilidad en las predicciones. Esto podría explicarse por el hecho de que los pilotos en estas posiciones suelen beneficiarse de condiciones más favorables, como estrategias más estables o una menor exposición a incidentes en carrera.

En cambio, a medida que las posiciones de clasificación empeoran, tanto la dispersión como la desviación media aumentan, evidenciando que al modelo le resulta más difícil capturar con exactitud el desempeño de estos pilotos en carrera. Este aumento de la desviación refleja que las condiciones y estrategias de los pilotos con peores clasificaciones son más variables y complejas de modelar. Además, podría estar relacionado con una mayor influencia de factores externos, como cambios en el clima, estrategias de adelantamiento o incluso incidentes inesperados. Esto deja en claro una tendencia recurrente en los resultados obtenidos.

PROBABILIDAD DE TERMINAR EN UNA BUENA POSICIÓN (1-10)

Esta gráfica muestra la **probabilidad de que los pilotos mantengan una buena posición** en carrera (definida como una posición final entre 1 y 10) en función de su posición de clasificación. De nuevo, se observa una tendencia clara: los pilotos que comienzan desde posiciones altas en la clasificación (más cercanas a 1) tienen una alta probabilidad de terminar en una buena posición en carrera, con valores cercanos al 100%. Sin embargo, esta probabilidad disminuye significativamente a medida que las posiciones de clasificación empeoran.

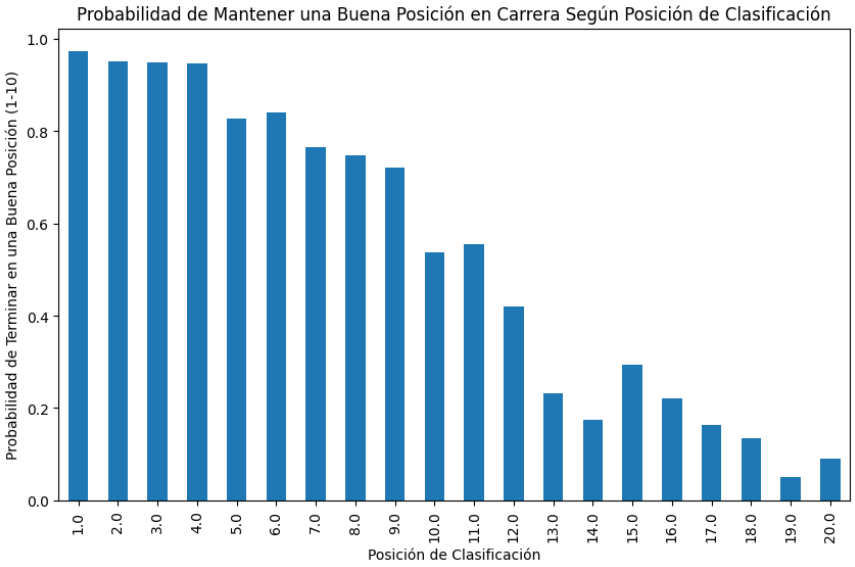

Ilustración 45: Probabilidad de mantener una buena posición en carrera

A partir de la posición 10 en clasificación, la probabilidad de mantener una buena posición en carrera cae de manera significativa, llegando a valores mínimos en posiciones finales de clasificación como la 19 o 20. Esto evidencia que el desempeño en carrera está altamente influenciado por la posición de salida, lo que sugiere que los pilotos con posiciones de clasificación bajas enfrentan mayores desafíos para lograr buenos resultados.

Estos resultados coinciden con las conclusiones obtenidas en las gráficas anteriores, reforzando la idea de que **el modelo es más preciso al predecir para pilotos que comienzan en posiciones altas de clasificación**. La tendencia observada, junto con la disminución de la probabilidad de mantener buenas posiciones en carrera desde posiciones más bajas, confirma que las condiciones de carrera para estos pilotos son más difíciles de modelar debido a factores como estrategias más arriesgadas o limitaciones inherentes al rendimiento.

DIAGRAMA DE DISPERSION DE POSICIONES REALES VS PREDICHAS EN CARRERA

Este estudio se generó como parte de un análisis complementario para explorar la relación entre las posiciones reales en carrera y las predicciones del modelo. Su propósito es visualizar cómo se ajustan las predicciones a los resultados reales, tomando como referencia la línea diagonal roja que indica una predicción perfecta. En esta gráfica, cada punto representa una predicción realizada por el modelo, con el eje X mostrando la posición real del piloto al final de la carrera y el eje Y mostrando la posición predicha. La línea roja punteada, denominada "**Línea Ideal**", refleja las predicciones perfectas, donde la posición real y la predicha coinciden.

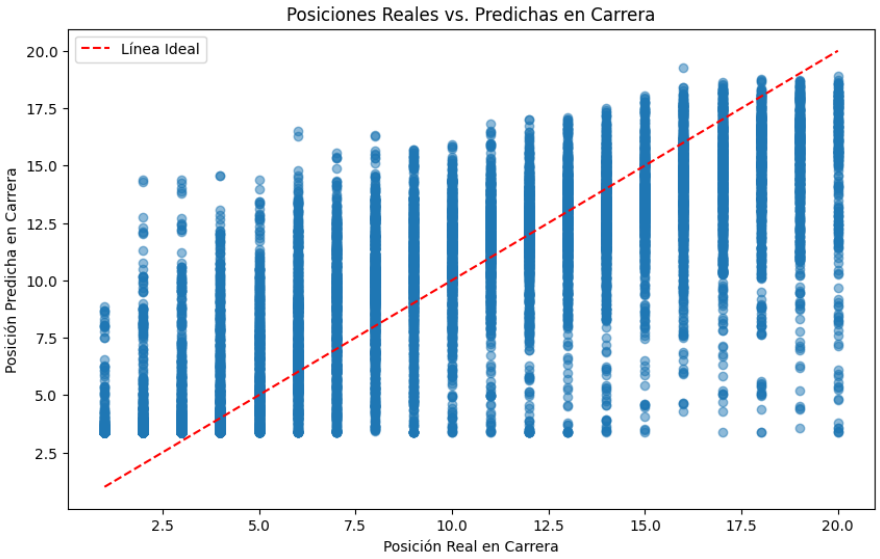

Ilustración 46: Relación entre Posiciones Reales y Predichas en Carrera

Los resultados reflejan una tendencia que coincide con los análisis anteriores: el modelo realiza predicciones más precisas para los pilotos que terminan en mejores posiciones finales, mientras que el error y la dispersión aumentan notablemente en las posiciones más bajas. Aunque la mayoría de los puntos están cercanos a la línea ideal, lo que indica un buen desempeño general del modelo, se observa cierta dispersión, especialmente en posiciones reales más altas (es decir, para pilotos que terminaron en peores posiciones). Esto sugiere que el modelo enfrenta más dificultades para predecir con precisión las posiciones finales de estos pilotos. Esto refuerza la confianza en las conclusiones previas, demostrando que las tendencias observadas no dependen de una única forma de visualización, sino que son consistentes y generales tanto en el modelo como en los datos analizados.

En definitiva, esta gráfica confirma que el modelo funciona de manera más sólida para predecir el desempeño de pilotos con buenas posiciones finales, pero encuentra mayores desafíos al intentar capturar el rendimiento de pilotos en los últimos puestos.

ERROR MEDIO ABSOLUTO (MAE) SEGÚN LA POSICIÓN DE CLASIFICACIÓN DE LOS PILOTOS

Tambien decidí calcular el MAE porque es una métrica fácil de interpretar que mide la desviación promedio entre las posiciones reales y las predichas. A diferencia del MSE, no amplifica errores grandes, lo que la hace más adecuada para evaluar un rango limitado de valores como las posiciones en carrera. Este cálculo me permitió identificar en qué rangos el modelo es más preciso y dónde enfrenta mayores dificultades, proporcionando una visión clara de su desempeño y de las áreas que podrían mejorarse

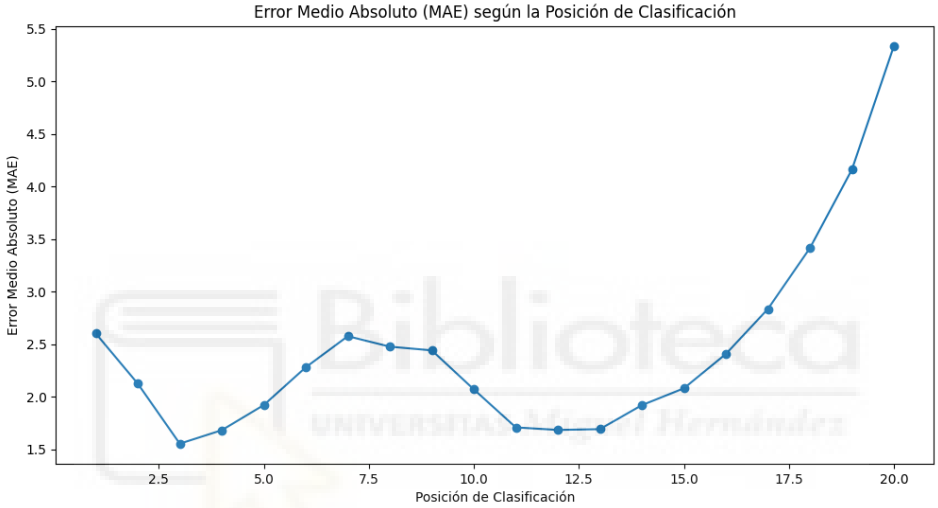

Ilustración 47: MAE según la posición de la clasificación

La gráfica de la ilustración 47 muestra el Error Medio Absoluto (MAE) del modelo en relación con la posición de clasificación de los pilotos. Cada punto en la línea representa el promedio de la diferencia absoluta entre las posiciones reales y las predichas para un grupo determinado de posiciones de clasificación.

Se puede observar que el error es notablemente menor para los pilotos que parten desde posiciones altas (cercanas a la 1), alcanzando su punto más bajo entre las posiciones 3 y 5, lo que indica una mayor precisión del modelo en estos casos. Sin embargo, a medida que las posiciones de clasificación empeoran, especialmente a partir de la posición 15, el MAE incrementa de forma considerable, alcanzando su valor más alto en la posición 20.

Al igual que las gráficas anteriores, esta visualización del MAE refuerza la conclusión de que el modelo tiene un mejor desempeño para predecir las posiciones finales de los pilotos

que clasifican en los primeros lugares. Aunque llegamos a la misma conclusión que con los resultados anteriores, este análisis era importante para evaluar el desempeño del modelo desde una perspectiva cuantitativa y no solo visual. Al calcular el MAE, hemos podido medir de forma concreta cómo varía la precisión del modelo en función de la posición de clasificación, lo que fortalece la confianza en los hallazgos observados en las gráficas previas.

PREDICCIÓN DE POSICIÓN EN CARRERA SEGÚN POSICIÓN DE CLASIFICACIÓN

La siguiente gráfica, junto con la tabla de resultados presentada más adelante, forma parte de un análisis que busca evaluar la precisión del modelo al predecir las posiciones finales de los pilotos en carrera en función de su posición de clasificación y otras características.

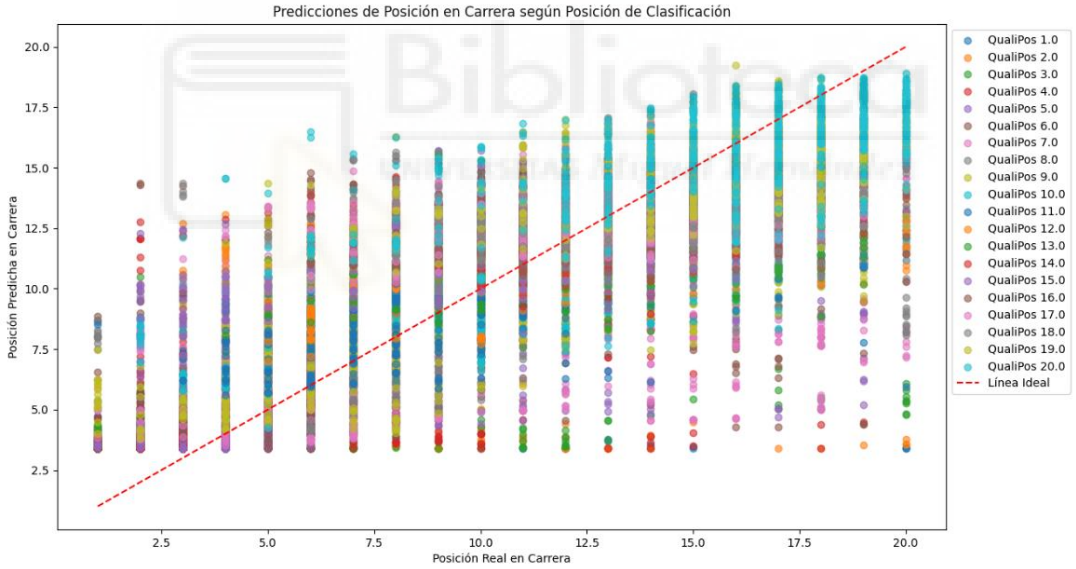

Ilustración 48: Relación entre Posición Real y Predicha en Carrera según la Posición de Clasificación

En la gráfica se representan las **posiciones reales** en el eje X y las **predicciones del modelo** en el eje Y, acompañadas de una línea roja que marca las predicciones perfectas, pudiendo asi **comparar ambos resultados**. Cada punto corresponde a un piloto y su color refleja su posición de clasificación inicial. Los puntos cercanos a la línea roja indican predicciones precisas, algo que es especialmente evidente en las mejores posiciones finales. Sin embargo, a medida que las posiciones reales aumentan (es decir, los

resultados son peores), se observa una mayor dispersión, reflejando que al modelo le cuesta más predecir con precisión estos casos.

Por otro lado, la tabla muestra ejemplos específicos, incluyendo el nombre del piloto, el circuito, la posición de clasificación, la posición real y la posición predicha por el modelo. Estos ejemplos muestran una comparación entre las posiciones reales y las predichas para un determinado piloto en un circuito especifico. Esto permite observar casos concretos, identificando situaciones en las que el modelo tiene un mejor o peor desempeño.

	Driver	Circuit	QualiPosition	Real Position	Predicted Position
0	PER	Mexico	11.0	7.0	9.545410
1	STR	Abu Dhabi	13.0	13.0	12.562973
2	NOR	Turkey	8.0	7.0	8.516489
3	NOR	Qatar	6.0	8.0	6.724189
4	GRO	Brazil	8.0	9.0	11.423935
5	LAT	Austria	16.0	19.0	16.840387
6	GAS	Azerbaijan	4.0	5.0	7.825212
7	STR	Spain	18.0	17.0	13.039619
8	MAG	Mexico	17.0	17.0	15.640249
9	HAM	Mexico	6.0	5.0	4.879415

Tabla 7: Predicciones del Modelo por Piloto y Circuito

Las columnas incluyen el nombre del piloto ("Driver"), el circuito en el que compitió ("Circuit"), su posición de clasificación ("QualiPosition"), la posición real en carrera ("Real Position") y la posición predicha en carrera por el modelo ("Predicted Position").

Además, cabe destacar que esta tabla de resultados se generó automáticamente a partir de un script en Python, que combina las predicciones del modelo con datos del conjunto de prueba, como el piloto, el circuito y la posición de clasificación. Los pilotos y circuitos que aparecen en la tabla fueron seleccionados aleatoriamente del conjunto de prueba, lo que permite analizar cómo se comporta el modelo en diferentes situaciones de manera representativa.

La tabla complementa este análisis al proporcionar detalles concretos de pilotos y circuitos, lo que permite explorar de manera más específica dónde el modelo logra predicciones más acertadas o encuentra mayores dificultades. Ambas herramientas coinciden en confirmar que el modelo es más fiable para pilotos con buenas posiciones iniciales y finales, mientras que presenta desafíos significativos al tratar los últimos puestos. Este análisis no solo refuerza las conclusiones obtenidas en visualizaciones

previas, sino que también valida que estas tendencias son consistentes y no dependen de una sola forma de evaluación.

En conjunto, las diferentes visualizaciones generadas no solo confirman las conclusiones ya obtenidas, sino que también resaltan la importancia de utilizar diversas perspectivas gráficas para validar el rendimiento del modelo. Aunque todas las gráficas muestran esencialmente el mismo patrón (mejor precisión en las predicciones para pilotos con buenas posiciones finales y mayores dificultades en las posiciones más bajas), analizarlas en conjunto permite entender mejor tanto las fortalezas como las limitaciones del modelo, además de los factores externos que podrían estar afectando su desempeño.

6.2.3.3 CONCLUSIÓN DEL MODELO 3

Este modelo demuestra un desempeño sólido al predecir las posiciones finales en carrera, particularmente para los pilotos que clasifican en posiciones altas. Esto indica que el modelo es capaz de identificar y capturar patrones claros en los datos para este grupo, ofreciendo predicciones consistentes y precisas. Sin embargo, su precisión disminuye de forma notable al tratar de predecir las posiciones finales de pilotos con clasificaciones iniciales bajas, lo cual puede atribuirse a una mayor incertidumbre y variabilidad en las estrategias de carrera y factores externos que afectan a estos pilotos.

En general, el modelo cumple adecuadamente con su objetivo en escenarios donde las condiciones son más predecibles, pero encuentra limitaciones al abordar las dificultades de los pilotos con peores posiciones de salida. Esto sugiere que, aunque el modelo proporciona información valiosa, existen áreas que podrían optimizarse para mejorar su capacidad de generalización en contextos más diversos.

6.2.4 MODELO 4: Impacto de la Estrategia de Neumáticos en el Rendimiento por Piloto

El objetivo principal del Modelo 4 es evaluar cómo distintas estrategias de carrera impactan en el rendimiento de los pilotos de Fórmula 1. Esto incluye analizar estrategias de paradas en boxes, el uso de diferentes compuestos de neumáticos, el impacto del desgaste de estos y la influencia de las condiciones de la pista. El enfoque es simular diversos escenarios para determinar cómo estas decisiones afectan la posición final de los pilotos, proporcionando una comprensión integral de los factores clave que pueden influir en el resultado de una carrera.

A lo largo del desarrollo del modelo, se realizaron múltiples simulaciones y visualizaciones para entender el comportamiento del rendimiento del piloto bajo diferentes condiciones. En la presente documentación se detallan todas las pruebas, cálculos y visualizaciones realizados, incluyendo las conclusiones obtenidas, tanto de los resultados exitosos como de aquellos que no mostraron variabilidad significativa.

6.2.4.1 OBJETIVOS ESPECIFICOS DEL MODELO 4

1. **Evaluar el Impacto de las Estrategias de Paradas en Boxes**: Determinar cuándo es el mejor momento para realizar una parada en boxes para minimizar la pérdida de posición y mejorar el rendimiento global de la carrera.

2. **Simular Estrategias de Paradas Múltiples**: Evaluar el impacto de realizar más de una parada en boxes durante la carrera y cómo se compara con estrategias de paradas únicas.

3. **Impacto del Desgaste de Neumáticos en el Rendimiento del Piloto**: Estudiar cómo el desgaste de los neumáticos afecta el rendimiento del piloto a lo largo de la carrera.

4. **Impacto de las Condiciones de la Pista en el Rendimiento del Piloto**: Analizar cómo diferentes estados de la pista (bandera amarilla, coche de seguridad, etc.) influyen en la posición final del piloto.

6.2.4.2 ARQUITECTURA DEL MODELO – VERSIÓN INCIAL: LSTM

En el desarrollo del Modelo 4, se exploraron dos versiones utilizando arquitecturas diferentes para abordar este modelo. En este apartado se describe en detalle la primera implementación realizada y se exponen las razones por las cuales se optó por un cambio en la arquitectura.

En la primera versión del modelo, se implementó una arquitectura basada en redes neuronales LSTM (Long Short-Term Memory). Esta elección inicial se justificó por la capacidad de las LSTM para capturar dependencias temporales en datos secuenciales, como los de las vueltas en una carrera. La arquitectura diseñada incluía:

1. **Capa de Entrada**: Procesaba las características de las vueltas, incluyendo tiempos de vuelta, desgaste de neumáticos, compuestos y estados de la pista.
2. **Capa LSTM**: Con 50 unidades para modelar relaciones temporales.
3. **Capa de Dropout**: Con una tasa del 20% para prevenir sobreajuste.
4. **Capa Densa**: Una única unidad de salida con activación lineal para predecir posiciones.
5. **Función de Pérdida y Optimización**: Se utilizó el MSE como métrica de error y el optimizador Adam.

Aunque los resultados iniciales mostraron cierto grado de coherencia con las dinámicas temporales del problema, se detectaron errores en las predicciones. Las gráficas generadas reflejaron inconsistencias en los patrones esperados, como una falta de sensibilidad a ciertos cambios en las estrategias de neumáticos. Esto llevó a la decisión de explorar una arquitectura diferente, la cual se explicará y detallará más adelante.

6.2.4.3 PROBLEMAS ENCONTRADOS EN EL DESARROLLO CON LSTM

Durante el desarrollo de la primera versión del Modelo 4, basada en una arquitectura LSTM, surgieron varios desafíos que afectaron a los resultados y llevaron a reconsiderar la arquitectura empleada. A continuación, se detallan los principales problemas encontrados:

1. **Inconsistencia en las Predicciones**

   Las gráficas generadas a partir de las predicciones del modelo mostraban comportamientos inesperados, como:

   - Falta de sensibilidad a cambios estratégicos significativos, como el impacto de las paradas en boxes o el desgaste de neumáticos.
   - Patrones irregulares en las posiciones predichas que no reflejaban adecuadamente las dinámicas esperadas en las carreras, como la pérdida de posiciones tras una parada en boxes y la posterior recuperación.

2. **Tiempos de Entrenamiento Prolongados**

   El entrenamiento del modelo fue considerablemente lento debido a la naturaleza secuencial de las LSTM. Esto dificultó la experimentación rápida con diferentes configuraciones y la búsqueda de mejores hiperparámetros.

3. **Limitaciones en la Captura de Relaciones Globales**

   Aunque las LSTM lograron capturar ciertas relaciones temporales locales, no pudieron modelar con precisión relaciones globales entre las vueltas, como el impacto acumulativo del desgaste de neumáticos o las dinámicas generales de la carrera.

4. **Falta de Interpretabilidad en las Predicciones**

   La naturaleza de "caja negra" de las LSTM dificultó entender cómo las variables de entrada estaban influyendo en las predicciones. Esto complicó el análisis de qué factores estratégicos tenían un mayor impacto en el rendimiento del piloto.

Estos problemas hicieron que el modelo LSTM no pudiera cumplir bien con los objetivos, como analizar las estrategias de neumáticos y las condiciones de la pista con precisión. Aunque al principio las LSTM parecían una buena opción, los inconvenientes que surgieron llevaron a probar un modelo Transformer, que resultó ser mucho más adecuado para este caso.

6.2.4.5 RESULTADOS DEL MODELO CON LSTM

EVOLUCIÓN DE LA PÉRDIDA DE ENTRENAMIENTO Y EVALUACIÓN

Durante el entrenamiento del modelo, se realizó un seguimiento de las pérdidas de entrenamiento y validación, tal como se muestra en la ilustración 49. Las curvas indican una reducción progresiva de la pérdida, con una convergencia entre ambas hacia el final del entrenamiento. Esto demuestra que el modelo fue capaz de aprender de manera efectiva, evitando problemas como el sobreajuste.

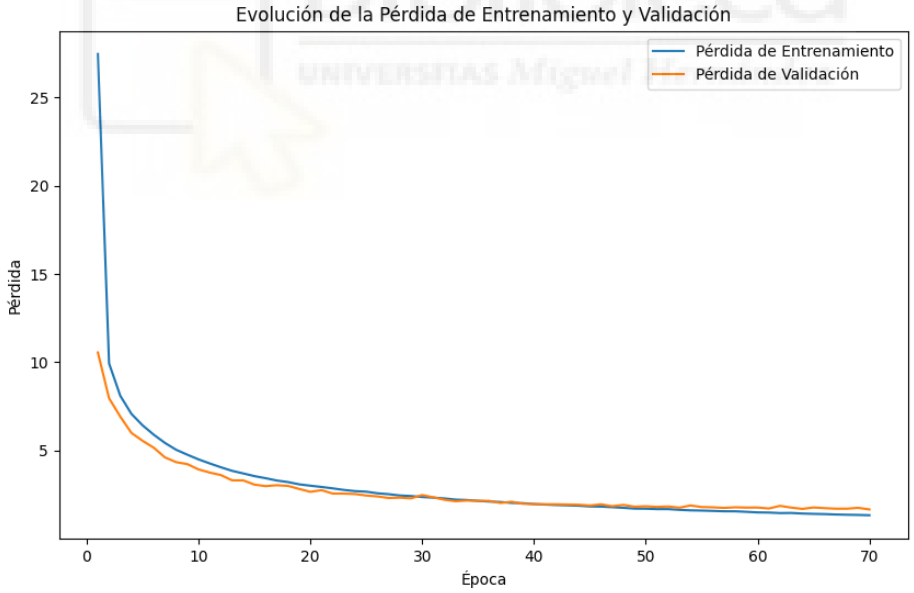

Ilustración 49: Curva de Pérdida durante el Entrenamiento y Validación del Modelo con LSTM

COMPARACIÓN ENTRE POSICIONES PREDICHAS Y REALES

En este análisis hemos comparado las posiciones reales (representadas en azul) con las predichas por el modelo (en naranja) para diferentes secuencias. En general, ambas líneas se mantienen bastante cercanas, lo que indica que el modelo ha sido capaz de capturar de manera adecuada el comportamiento observado en la realidad.

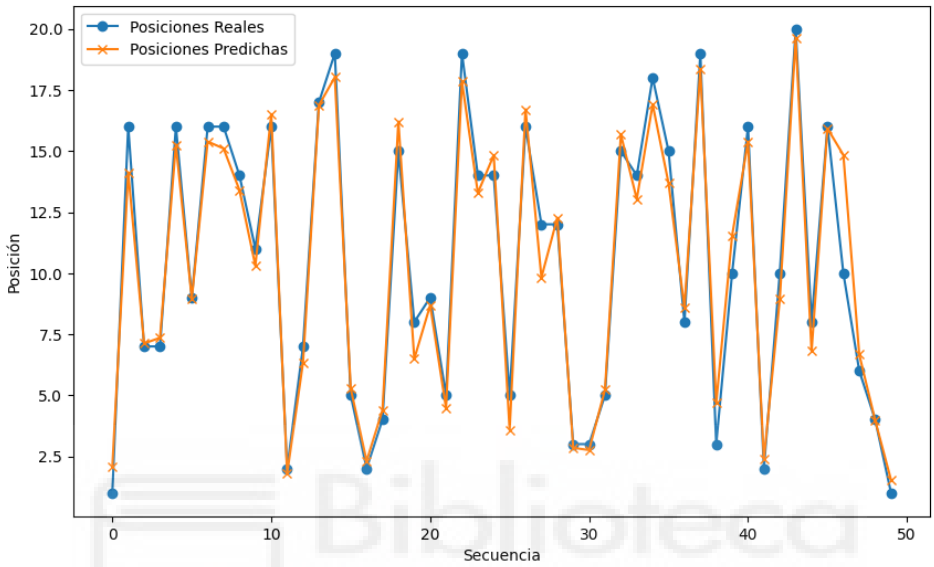

Ilustración 50: Comparación entre Posiciones Reales y Predichas por el Modelo con LSTM

Aunque existen algunos puntos en los que las predicciones presentan desviaciones respecto a las posiciones reales, esto es comprensible debido a la dificultad de modelar carreras de Fórmula 1, donde intervienen numerosos factores. A pesar de estas diferencias puntuales, el modelo sigue correctamente las tendenciales generales: cuando las posiciones reales aumentan o disminuyen, las predicciones reflejan cambios similares. Esto evidencia que el modelo ha logrado comprender la dinámica general de la competición.

SIMULACIÓN DE CARRERA CON Y SIN PARADA EN BOXES

Tambien se ha analizado el desempeño de un piloto bajo dos estrategias distintas: una sin paradas en boxes (línea azul) y otra con una parada estratégica en la vuelta 10 (línea roja punteada).

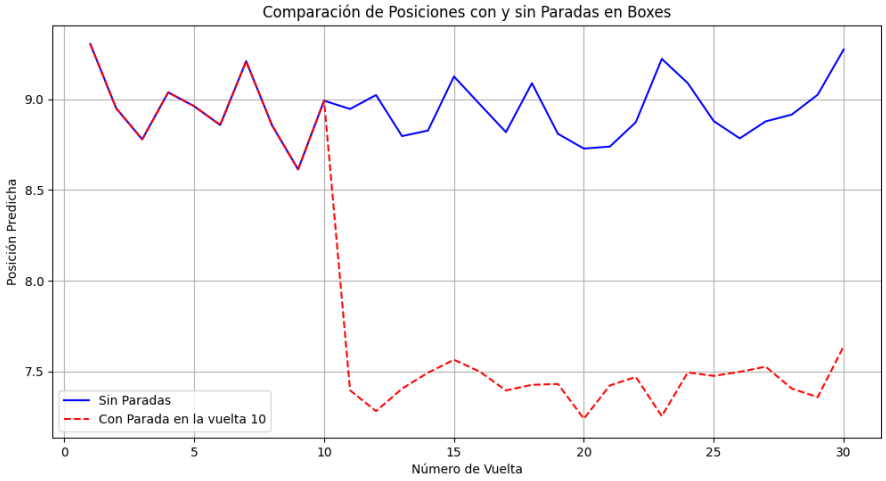

Ilustración 51: Impacto de las Paradas en Boxes - LSTM

Sin Parada:

La línea azul muestra el desempeño del piloto durante la carrera sin realizar ninguna parada en boxes empezando desde la posición 10, donde se observa que la posición predicha varía a lo largo de las vueltas debido a los factores de cambio que se introdujeron para simular condiciones como el tráfico en pista y el rendimiento del vehículo. Este comportamiento refleja la competencia entre los pilotos en un escenario donde no se realizan ajustes estratégicos importantes

Parada en la vuelta 10

La línea roja punteada representa lo que sucede al realizar una parada estratégica en la vuelta 10. A diferencia de lo esperado, el gráfico muestra que la posición mejora inmediatamente después de la parada, acercándose al valor 1 (habiendo entrado a boxes mientras el piloto se encontraba en la posición 9), lo que no corresponde con la realidad de una carrera. En situaciones reales, la parada debería causar una pérdida inicial de posiciones, mostrando valores más altos, para después recuperarlas con el paso de las vueltas gracias al uso de neumáticos nuevos. Este resultado erróneo fue una de las razones que llevó a probar una versión mejorada del modelo usando un Transformer Encoder, que

permitió captar mejor las relaciones entre las vueltas y representar de forma más real lo que ocurre en una carrera.

IMPACTO DEL DESGASTE DE NEUMÁTICOS EN EL RENDIMIENTO

Para el siguiente estudio, se analizó el desgaste de los neumáticos en detalle para entender cómo afecta el desgaste de los neumáticos a las posiciones predichas.

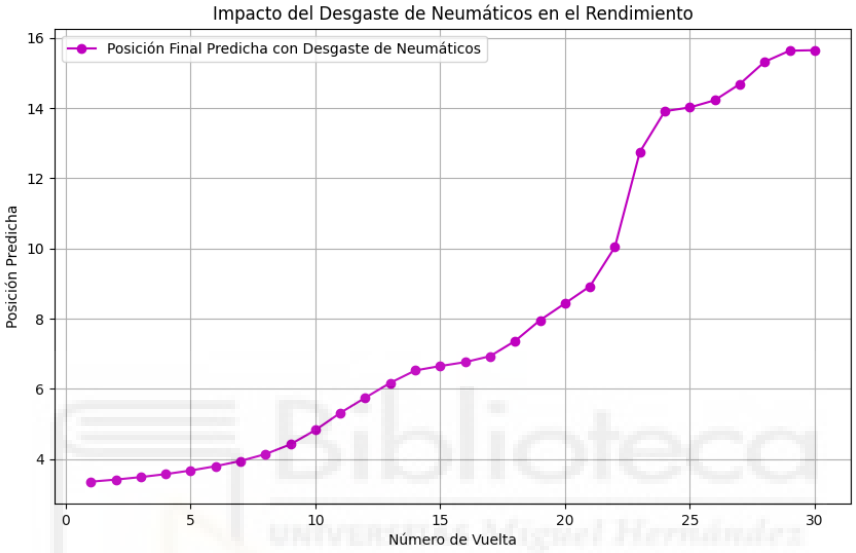

Ilustración 52: Evolución de la Posición Final Predicha en Función del Desgaste de Neumáticos - LSTM

Podemos observar en la gráfica un incremento evidente en la posición predicha, lo que representa una caída en el rendimiento del piloto a medida que avanza la carrera, algo lógico debido al desgaste progresivo de los neumáticos. No obstante, si la pendiente del incremento es demasiado pronunciada, podría indicar que el efecto del desgaste ha sido sobrerrepresentado o no está correctamente ajustado en el modelo.

Un detalle interesante es que alrededor de la vuelta 20 se detecta un "punto crítico", donde el deterioro en la posición se acelera notablemente. Este fenómeno podría ser investigado más a fondo para identificar estrategias óptimas de cambio de neumáticos.

IMPACTO DE LAS CONDICIONES DE LA PISTA EN EL RENDIMIENTO DEL PILOTO

En este apartado se examinan las diferentes condiciones de la pista y su impacto en la posición final predicha de los pilotos. Cada barra de la gráfica representa una condición específica:

- Condición 1: **Pista despejada**. No hay incidentes, banderas ni limitaciones. Los pilotos pueden competir sin restricciones.
- Condición 2: **Bandera amarilla**. Se ha producido un incidente y los pilotos deben reducir la velocidad en una parte de la pista, sin adelantamiento permitidos en la zona afectada.
- Condición 4: **Coche de seguridad**. Un coche de seguridad está en pista, y todos los pilotos deben reducir la velocidad y alinearse detrás del coche. Está prohibido adelantar.
- Condición 5: **Bandera roja**. La carrera se suspende temporalmente debido a un incidente grave o condiciones peligrosas. Todos los pilotos deben detenerse y la carrera se pausa hasta que sea seguro reanudarla.
- Condición 6: **Coche de seguridad virtual (VSC) desplegado.** Todos los pilotos deben reducir la velocidad a un delta de tiempo específico para mantener una velocidad segura, sin adelantamientos.
- Condición 7: **Fin del coche de seguridad virtual (VSC).** Esta condición indica el momento en el que el coche de seguridad virtual se retira y se reanuda la carrera con normalidad.

Sin embargo, no se incluye la **Condición 3**, ya que, según la documentación de FastF1, esta no tiene una definición clara o un significado especifico. Dado que desconozco su naturaleza, he optado por omitirla en el análisis principal para evitar posibles alteraciones en los datos o interpretaciones erróneas.

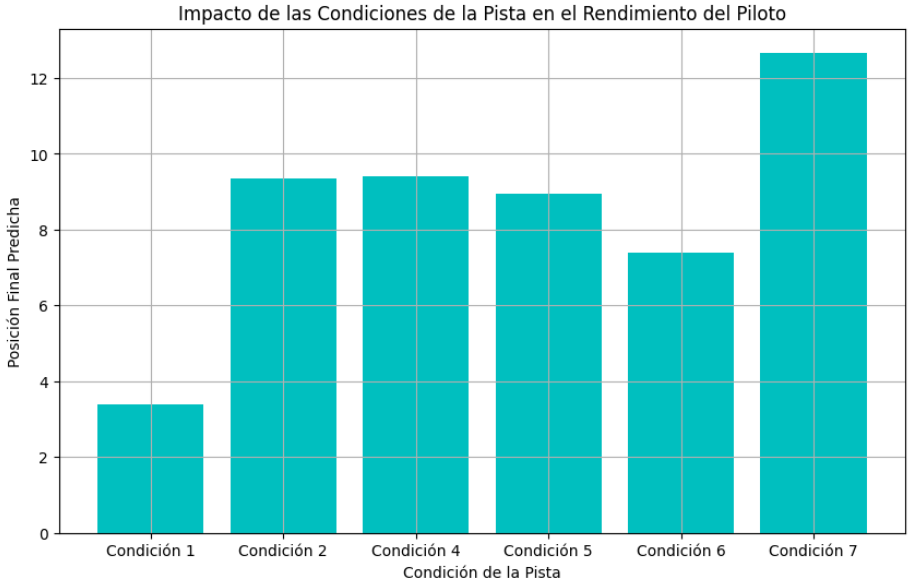

Ilustración 53: Efecto de las Condiciones de la Pista en las Posiciones Finales Predichas - LSTM

Podemos observar como la **Condición 1 (Pista Despejada)** muestra el mejor rendimiento con posiciones finales predichas más cercana a la parte superior de la clasificación (valores más bajos en la gráfica). Esto es esperado, ya que una pista despejada permite a los pilotos mantener un ritmo constante y sin interrupciones. En cambio, con la presencia de una **bandera amarilla (Condición 2)** se observa un deterioro en el rendimiento en comparación con la condición 1. Como he mencionado anteriormente, las banderas amarrillas suelen reducir la velocidad en ciertos sectores, lo que afecta el tiempo por vuelta, y por tanto la posición final. Similar a la condición 2, pero con un impacto ligeramente mayor tenemos la **Condición 4** (**Coche de Seguridad**).

En el caso de una **Bandera Roja (Condición 5),** esta muestra un impacto algo comparable al del coche de seguridad. Como las banderas rojas suelen detener la carrera, la estrategia y el ritmo se pueden ver afectados, especialmente si ocurre cerca del final.

La **Condición 6 (Coche de Seguridad Virtual)** aunque tiene un impacto menor que el coche de seguridad completo (situación de bandera roja), aún se refleja una pérdida en el rendimiento debido a la limitación de velocidad en ciertas zonas.

Por último, la **Condición 7 (Fin del Coche de Seguridad Virtual)** tiene el peor impacto, con las posiciones finales más desfavorables. Esto podría estar relacionado con una

estrategia mal ajustada después de la reanudación, donde los pilotos intentan recuperar tiempo perdido per enfrentan mayor competencia.

6.2.4.6 ARQUITECTURA DEL MODELO – VERSIÓN FINAL: TRANSFORMER

Como mencioné anteriormente, tras analizar las limitaciones del modelo basado en LSTM, decidí implementar una versión mejorada utilizando un **Transformer Encoder**. Esta arquitectura es conocida por su capacidad para capturar relaciones complejas entre los elementos de una secuencia, gracias a su mecanismo de autoatención, lo que la hace adecuada para este tipo de problemas.

El modelo implementado cuenta con los siguientes componentes principales:

1. **Capa de Entrada:** Esta capa transforma las características de las vueltas (como tiempos de sectores, desgaste y compuestos de neumáticos, velocidades y estados de pista) en un espacio de representación de mayor dimensión mediante una capa densa. Este paso es fundamental para preparar los datos para las capas posteriores.

2. **Capa de Transformer Encoder:** Aquí implemente dos capas "TransformerEncoderLayer", cada una con:
    - **Autoatención Multi-Cabezal:** Utilicé cuatro cabezas de atención, lo que permite al modelo identificar relaciones importantes entre diferentes vueltas dentro de la secuencia, como la influencia de una parada en boxes en vueltas posteriores.
    - **Capa Feed-Forward:** Incluye una red densa que refuerza las representaciones aprendidas después del módulo de autoatención.
    - **Dropout y Normalización:** Para evitar sobreajustes y estabilizar el entrenamiento, apliqué una tasa de dropout del 30% y normalización en cada subcapa.

3. **Capa de Predicción:** La salida correspondiente al último paso temporal de la secuencia (última vuelta analizada) se extrae y pasa por una capa densa que genera una predicción escalar, representando la posición estimada del piloto.

4. **Función de Pérdida y Optimización:** Elegí la función de pérdida MSE para medir el error en las predicciones, al tratarse de un problema de regresión. Además, utilicé el optimizador Adam con regulación L2 para mejorar la estabilidad del entrenamiento.

Hiperparámetros del Modelo

En la siguiente tabla se presenta un resumen de los hiperparámetros utilizados en esta versión son los siguientes:

Hiperparámetros	**Valor**	**Descripción**
Dimensión oculta	32	Espacio interno de representación en cada capa del Transformer
Número de cabezas	4	Número de cabezas de atención en cada capa del Transformer Encoder
Número de capas	2	Número de capas TransformerEncoderLayer utilizadas
Dropout	0.3	Tasa de dropout aplicada para prevenir el sobreajuste
Función de pérdida	MSE	Métrica utilizada para evaluar el error en la tarea de regresión
Optimización	Adam	Optimizador empleado con regularización L2

6.2.4.7 PROBLEMAS ENCONTRADOS EN EL DESARROLLO CON TRANSFORMER

Durante la implementación de esta versión del modelo tambien tuve que enfrentarme a varios problemas importantes que requirieron ajustes en la configuración y el diseño del modelo. A pesar de estas dificultades, cada reto fue una oportunidad para refinar la arquitectura y mejorar su rendimiento. A continuación, se explican los principales problemas y las soluciones aplicadas:

1. **Altos Tiempos de Entrenamiento:** Uno de los problemas más común fue el tiempo que tomaba entrenar el modelo, ya que, la naturaleza compleja del Transformer, combinada con el volumen de los datos procesados, ralentizaba las iteraciones necesarias para ajustar los hiperparámetros y evaluar los resultados.

   Para resolver este problema lo que hice fue reducir la dimensión oculta, el número de capas y el número de cabezas de atención. Estos ajustes mejoraron significativamente la velocidad de entrenamiento sin comprometer la capacidad predictiva del modelo.

2. **Problemas con la Configuración del Dispositivo:** Durante las primeras pruebas, encontré dificultades para asignar correctamente el modelo y los datos al dispositivo adecuado (CPU o GPU), lo que generó errores que interrumpieron el entrenamiento y requirieron ajustes en la configuración inicial.

   Solucioné este problema implementando una configuración explicita para detectar automáticamente el dispositivo disponible (cuda o cpu) y asegurar que tanto los datos como el modelo se asignaran correctamente, resolviendo el problema de manera efectiva.

3. **Manejo de Secuencias de Diferente Longitud:** Al procesar las vueltas de carreras como secuencias, me di cuenta de que no todas las entradas tenían la misma longitud, y esto generaba inconsistencias al procesar las secuencias al Transformer, que requiere que las entradas sean uniformes.

Para solucionar este problema implemente un esquema para truncar las secuencias más largas y rellenar las más cortas con valores neutros (padding). Esto garantizó que todas las secuencias tuvieran la misma longitud permitiendo que el modelo las procesara correctamente.

6.2.4.8 RESULTADOS DEL MODELO CON TRANSFORMER ENCONDER

En este apartado se explicarán los resultados obtenidos con la versión del modelo desarrollada con Transformer Encoder, y se destacarán las diferencias más relevantes respecto a la versión anterior basada en LSTM, resaltando las ventajas del Transformer en este contexto.

EVOLUCIÓN DE LA PÉRDIDA DE ENTRENAMIENTO Y EVALUACIÓN

En esta gráfica podemos ver como a diferencia de la LSTM, el Transformer alcanza una convergencia más rápida, lo cual era esperable dada su capacidad para procesar las relaciones dentro de una secuencia de forma simultánea en lugar de hacerlo de manera secuencial.

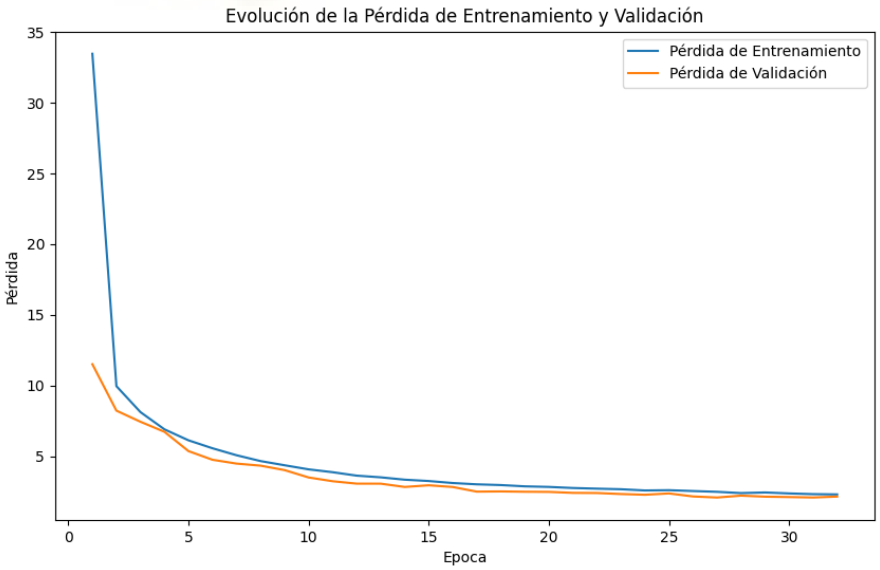

Ilustración 54: Curva de Pérdida durante el Entrenamiento y Validación del Modelo con Transformer

El Transformer presenta una disminución significativa de la pérdida tanto en entrenamiento como en validación durante las primeras 10 épocas, estabilizándose entre

las épocas 15 y 20. Este comportamiento refleja su capacidad para captar relaciones globales en las secuencias de manera eficiente, sin necesidad de tantas iteraciones como la LSTM.

Podemos apreciar como igual que con la LSTM, la pérdida de validación se mantiene cercana a la de entrenamiento a lo largo del proceso, lo que indica una buena generalización del modelo. Esto sugiere que el Transformer no solo aprende patrones relevantes, sino que además evita sobre ajustarse a los datos. En cuanto a la estabilización de la pérdida, esta ocurre de forma temprana, con valores bajos y consistentes a partir de la época 20. Esto contrasta con la LSTM, que mostró una disminución más progresiva y requirió más épocas para estabilizarse.

La siguiente tabla muestra un resumen de las diferencias más importantes:

Criterio	**Transformer**	**LSTM**
Velocidad de Convergencia	Se estabiliza bastante antes, sobre la época 15.	Necesita entre 30 y 40 épocas para alcanzar un comportamiento estable.
Relaciones Capturadas	Captura relaciones globales en toda la secuencia de manera simultánea.	Procesa las relaciones paso a paso, lo que aumenta el tiempo de entrenamiento.
Consistencia entre Pérdidas	La pérdida de validación sigue de cerca a la de entrenamiento con menos iteraciones.	La pérdida de validación tambien sigue de cerca a la de entrenamiento, pero requiere más iteraciones para lograrlo.

Tabla 8: Comparación entre Transformer y LSTM

COMPARACIÓN ENTRE POSICIONES PREDICHAS Y REALES

En este caso, aunque el Transformer presenta un buen rendimiento, la gráfica de la LSTM [ilustración 50] parece mostrar un ajuste más preciso en términos globales, con predicciones que se alinean más estrechamente con las posiciones reales en la mayoría de los casos, pues las desviaciones entre las posiciones predichas y reales son más notorias en algunos puntos en el Transformer, especialmente en secuencias de alta variabilidad. En este aspecto, la LSTM parece ofrecer predicciones más consistentes.

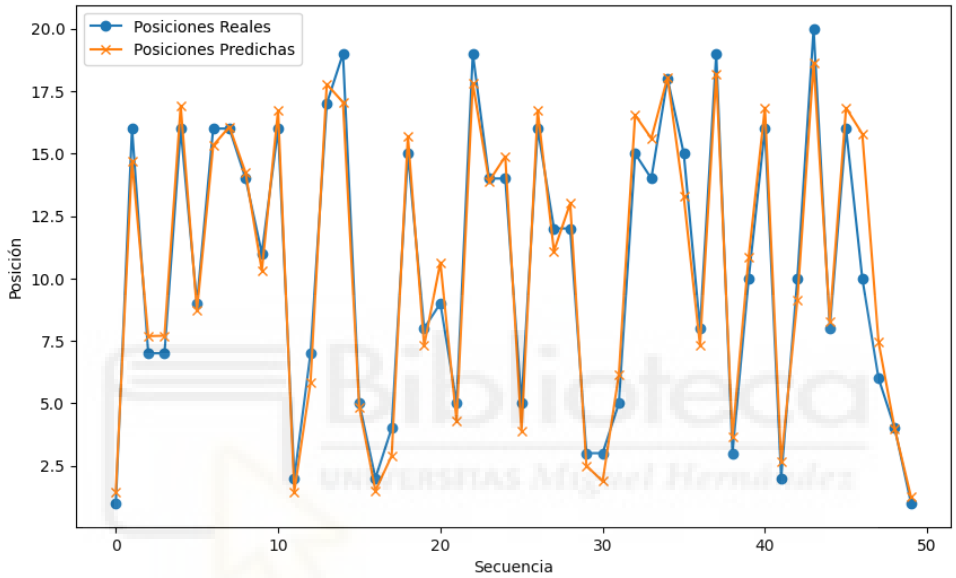

Ilustración 55: Comparación entre Posiciones Reales y Predichas por el Modelo con Transformer

SIMULACIÓN DE CARRERA CON Y SIN PARADA EN BOXES

A diferencia de los resultados obtenidos con la LSTM, los cuales no eran coherentes, he podido sacar conclusiones mucho más precisas y realistas con esta versión del modelo desarrollada con Transformer Encoder.

Se observa que, tras realizar la parada en la vuelta 10 (línea roja punteada), hay una pérdida inmediata de posiciones, lo cual es coherente con la dinámica real de una carrera. Posteriormente, el piloto debería comenzar a recuperar posiciones debido al beneficio de los neumáticos nuevos, demostrando un mejor rendimiento en las vueltas posteriores. En cambio, la línea azul (sin paradas) se mantiene estable, reflejando que el rendimiento sin intervención estratégica (como las paradas) sigue un comportamiento predecible.

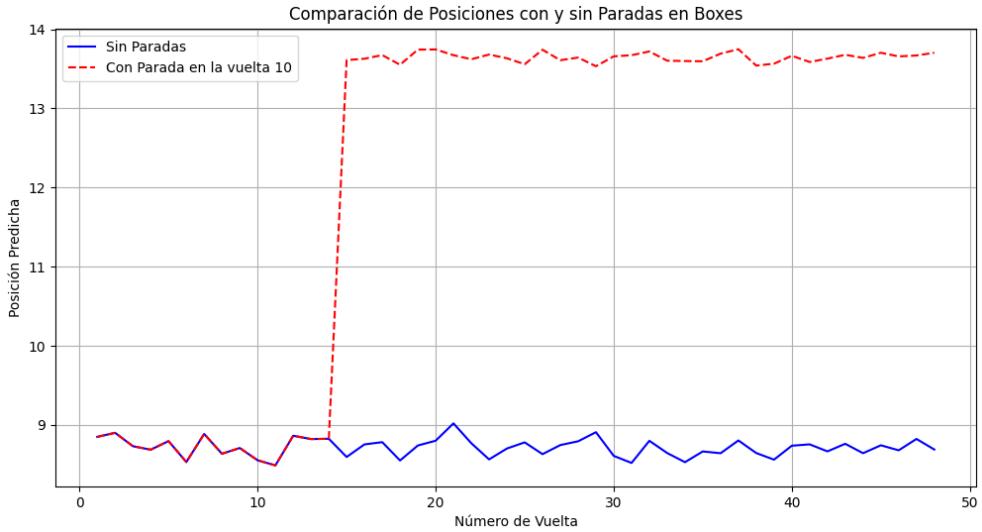

Ilustración 56: Impacto de las Paradas en Boxes - TRANSFORMER

Las tendencias observadas son lógicas y alineadas con lo que sucede en un contexto real de carrera. Esto demuestra que el Transformer capta mejor las relaciones globales y locales en los datos, proporcionando una representación más precisa. Esta fue una de las razones principales por la que se optó probar esta versión mejorada del modelo.

IMPACTO DEL DESGASTE DE NEUMÁTICOS EN EL RENDIMIENTO

La siguiente gráfica muestra como el desgaste de los neumáticos afecta a la posición final predicha de un piloto a lo largo de 50 vueltas.

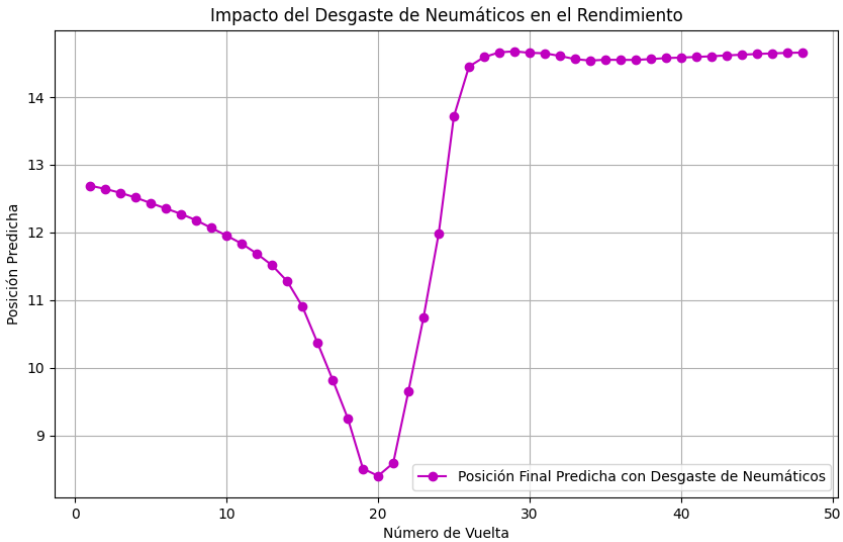

Ilustración 57: Evolución de la Posición Final Predicha en Función del Desgaste de Neumáticos - Transformer

Al inicio de la carrera, el modelo predice una posición relativamente estable alrededor de la posición 12. Sin embargo, conforme avanza el número de vueltas, el desgaste de los neumáticos comienza a tener un impacto progresivo en el rendimiento. Vemos también como en torno a la vuelta 20, la **posición predicha mejora** considerablemente, alcanzando un punto óptimo alrededor de la posición 9. Este descenso en el valor de la posición (aumento de posición en la F1) podría estar relacionado con una simulación de condiciones favorables, como un breve período de mejor gestión del desgaste o circunstancias en pista que benefician al piloto.

A partir de la vuelta 25, el modelo muestra un aumento pronunciado en la posición predicha, lo que indica un claro **deterioro en el rendimiento del piloto** debido al desgaste acumulativo de los neumáticos. Este comportamiento resalta cómo la degradación afecta considerablemente a la capacidad de mantener un buen ritmo.

Finalmente, en las últimas vueltas, la posición predicha se **estabiliza** alrededor de la posición 14. Esto sugiere que, aunque el deterioro es severo, el modelo identifica un punto en el que las condiciones no empeoran significativamente más allá de un cierto límite.

Las **diferencias clave con respecto** al modelo basado en **LSTM** es que el modelo basado en Transformer muestra cambios más dinámicos en el rendimiento a lo largo de la carrera, mientras que el LSTM genera un resultado más plano y uniforme, sin reflejar tantas variaciones. Por otro lado, el Transformer demuestra una mayor sensibilidad a los efectos acumulativos del desgaste de los neumáticos, identificando claramente momentos clave de mejora y deterioro, algo que el LSTM no logró captar de manera tan evidente.

La gráfica obtenida con esta versión del modelo captura mejor las fluctuaciones y los puntos de inflexión que se esperan en un entorno tan competitivo como una carrera de Fórmula 1.

IMPACTO DE LAS CONDICIONES DE LA PISTA EN EL RENDIMIENTO DEL PILOTO

Por último, voy a comparar esta gráfica que al igual que la vista en la versión LSTM de este modelo, muestra el impacto de las distintas condiciones de la pista en la predicción de la posición final del piloto.

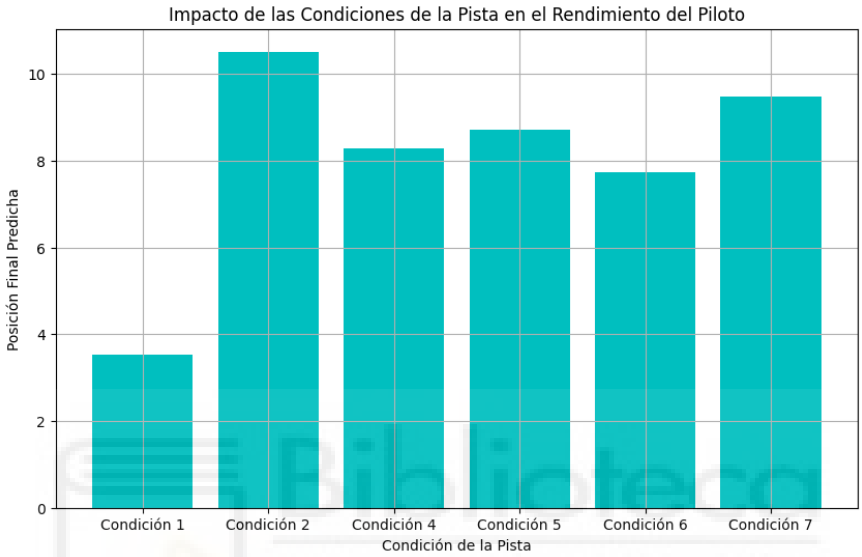

Ilustración 58: Efecto de las Condiciones de la Pista en las Posiciones Finales Predichas - Transformer

Lo primero que vemos es que, en este modelo la condición 1 (pista despejada) presenta un valor bajo en la predicción de la posición final, mientras que la condición 7 (fin del coche de seguridad virtual) muestra un valor significativamente alto. Este patrón es similar al observado con LSTM, aunque la magnitud de la predicción para la condición 7 parece ser más pronunciada en el Transformer, lo que podría indicar una mayor sensibilidad del modelo a esta condición específica. En el caso de la condición 2 (bandera amarilla), la posición predicha alcanza un pico más alto que con LSTM, mientras que las condiciones intermedias (4, 5 y 6) muestran una distribución de predicciones más uniforme, con barras relativamente cercanas entre sí.

En conclusión, el modelo Transformer demuestra ser más adecuado para este análisis, ya que refleja de manera más detallada y realista cómo las distintas condiciones de la pista afectan al rendimiento del piloto. Mientras que el LSTM ofrece un panorama más general, el Transformer es capaz de capturar las variaciones clave y proporcionar información más útil para la interpretación y toma de decisiones estratégicas.

6.2.4.9 CONCLUSIÓN DEL MODELO 4

A partir de los resultados y comparaciones expuestos, se puede concluir que la arquitectura basada en el Transformer Encoder proporciona ventajas significativas frente a la LSTM en la predicción de posiciones durante una carrera de Fórmula 1. En primer lugar, el Transformer converge de manera más rápida, tal como lo demuestra la estabilización temprana de la pérdida en entrenamiento y validación. Además, su capacidad de procesar simultáneamente las relaciones en la secuencia le permite capturar con mayor precisión los cambios y fluctuaciones en factores importantes, como el desgaste de los neumáticos, las paradas en boxes y las condiciones de la pista.

Aunque la LSTM logra un ajuste puntual más preciso en ciertos tramos, la visión global y la eficiencia del Transformer lo convierten en una opción más sólida para la toma de decisiones estratégicas en escenarios complejos. En definitiva, el modelo basado en Transformer Encoder ofrece un comportamiento más realista y coherente, lo que hace que sea una alternativa superior para el análisis y la optimización del rendimiento en carreras de Fórmula 1.

CÁPITULO 7: CONCLUSIONES Y TRABAJOS FUTUROS

7.1 Conclusiones

Este trabajo de fin de grado ha sido una experiencia enriquecedora que combinó conceptos teóricos avanzados con aplicaciones prácticas en el ámbito de la Fórmula 1, un deporte que representa la máxima expresión de la ingeniería y el análisis de datos. Durante el desarrollo, se enfrentaron diversos retos relacionados con la implementación de modelos de machine learning, destacando la importancia de los datos de telemetría como base para predecir el rendimiento de pilotos y coches.

A lo largo del proyecto, se desarrollaron cuatro modelos predictivos enfocados en aspectos clave del rendimiento en carrera, como la predicción de posiciones finales, el impacto de las condiciones climáticas, la influencia de la posición de clasificación y el papel estratégico de las paradas en boxes. Cada modelo presentó desafíos específicos, desde la falta de datos en ciertas áreas hasta problemas con el ajuste de hiperparámetros y la elección de arquitecturas adecuadas, que incluyeron desde redes MLP hasta redes Transformer. Afrontar estos retos requirió no solo un análisis técnico detallado, sino también adaptabilidad y un enfoque iterativo en la metodología de trabajo.

El proyecto también permitió aplicar diversas herramientas tecnológicas, como Python, Google Colab, Visual Studio Code y bibliotecas especializadas como FastF1, Pandas, NumPy, Matplotlib, PyTorch y Scikit-learn. Estas herramientas facilitaron tanto la extracción y procesamiento de los datos para la creación y evaluación de modelos sólidos.

Asi mismo, el aprendizaje adquirido en la implementación de machine learning en un caso real ha sido invaluable. Permitió desarrollar habilidades para identificar datos relevantes, manejar grandes volúmenes de información y optimizar modelos con técnicas como el balanceo de clases y el ajuste de hiperparámetros. Más allá de los números, este proyecto demostró cómo el machine learning puede convertir datos complejos en herramientas predictivas con aplicaciones prácticas en la toma de decisiones estratégicas.

Desde una perspectiva personal y profesional, este trabajo marcó un importante crecimiento en el desarrollo de habilidades técnicas, el manejo de herramientas avanzadas y la capacidad para resolver problemas complejos. También resaltó la relevancia de una

planificación detallada, como quedó reflejado en el diagrama de Gantt, y la importancia de mantener una visión clara de los objetivos incluso frente a imprevistos.

En conclusión, este proyecto no solo cumplió los objetivos planteados, sino que dejó una base sólida para futuras investigaciones. La experiencia adquirida confirma el poder transformador del machine learning y la importancia de la innovación en la ingeniería de datos en contextos tan competitivos y dinámicos como el de la Fórmula 1.

7.2 Futuros trabajos

En futuros trabajos, sería importante centrarse en mejorar el **Modelo 1**, ya que sus resultados no fueron satisfactorios. El modelo tuvo problemas para capturar la relación entre las posiciones pasadas de los pilotos y sus posiciones finales en carrera, lo que indica que no aprovechó bien esta variable como elemento clave para las predicciones. Una posible mejora sería explorar nuevas arquitecturas, como modelos que integren mecanismos de atención o enfoques específicos para datos secuenciales. Además, incluir variables adicionales que aporten más contexto, como estrategias de equipo o condiciones específicas de cada carrera, podría ayudar a mejorar el rendimiento del modelo.

Por otro lado, el **Modelo 2** tuvo una principal limitación que fue la falta de datos meteorológicos disponibles, lo que dificultó analizar el impacto real de las condiciones climáticas en el rendimiento de los pilotos. Una línea de trabajo futura sería recopilar un conjunto de datos más completo, incluyendo información como temperatura de la pista, humedad o velocidad del viento. Esto permitiría desarrollar un modelo más robusto y preciso en este ámbito.

Finalmente, todos los modelos presentados podrían beneficiarse de una revisión general para mejorar tanto su precisión como su eficiencia computacional. Esto incluiría ajustes en las arquitecturas existentes, pruebas con diferentes técnicas de preprocesamiento y la experimentación con optimizadores avanzados para reducir los tiempos de entrenamiento. Además, validar los modelos con conjuntos de datos más variados o de temporadas adicionales de Fórmula 1 sería clave para garantizar una mejor generalización y utilidad práctica.

CÁPITULO 8: BIBLIOGRAFÍA

8.1 Fuentes Bibliográficas

[1] Liberty Media, *F1.svg*, 2017. https://es.m.wikipedia.org/wiki/Archivo:F1.svg (accedido el 3 de julio de 2024)

[2] Wikipedia, *Logo oficial de Ferrari* https://es.wikipedia.org/wiki/Scuderia_Ferrari#/media/Archivo:Ferrari-badge.jpg (accedido el 3 de julio de 2024)

[3] Infobae, *¿Qué es la carrera sprint que Franco Colapinto correrá en el GP de Estados Unidos?*, 2024. https://www.infobae.com/deportes/2024/10/19/que-es-la-carrera-sprint-que-franco-colapinto-correra-en-el-gp-de-estados-unidos-los-cambios-para-esta-temporada/ (accedido el 3 de julio de 2024)

[4] Federación Internacional del Automóvil (FIA*), "FIA 2024 Formula 1 Technical Regulations,"*, 2023. https://www.fia.com/sites/default/ (accedido el 4 de julio de 2024)

[5] Federación Internacional del Automóvil (FIA), *"FIA 2024 Formula 1 Sporting Regulations,"*, 2023. https://www.fia.com/sites/default/files/fia_2024_formula_1_sporting_regulations_-_issue_1_-_2023-09-26.pdf (accedido el 4 de julio de 2024)

[6] Juanjo Sáez, "*Qué es el Parc Fermé o Parque cerrado en la F1? ¿Cómo funciona?*", 2024. https://es.motorsport.com/f1/news/parc-ferme-parque-cerrado-que-es-como-funciona/10562062/ (accedido el 4 de julio de 2024).

[7] Autor Desconocido, Physics of Formula 1, "*Modelling DRS*". https://physicsofformula1.wordpress.com/modelling-drs/ (accedido el 6 de julio de 2024)

[8] Coursera, *"History of Artificial Intelligence: A Timeline of Artificial Intelligence,"*, 2024. https://www.coursera.org/articles/history-of-ai (accedido el 6 de julio de 2024)

[9] Edureka, *"Top 10 data Sciencia Applications with Real Life Examples"*, 2024. https://www.edureka.co/blog/data-science-applications/ (accedido el 9 de julio de 2024)

[10] IBM, *"Artificial Intelligence vs. Machine Learning vs. Deep Learning vs. Neural Networks: What´s the difference?"*, 2023. https://www.ibm.com/think/topics/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks (accedido el 14 de julio de 2024)

[11] IBM, *"What is Deep Learning?"*, 2024. https://www.ibm.com/topics/deep-learning (accedido el 14 de julio de 2024)

[12] IBM, "*What Is Machine Learning and Why It Matters?*" https://www.ibm.com/cloud/learn/machine-learning (accedido el 14 de julio 2024)

[13] Coursera, "*Supervised vs Unsupervised Learning*", 2023. https://www.coursera.org/articles/supervised-unsupervised-learning (accedido el 14 de julio de 2024).

[14] Gliese, "*Ejemplo de Clustering con K-Means en Python*", 2019 https://exponentis.es/ejemplo-de-clustering-con-k-means-en-python (accedido el 18 de julio de 2024)

[15] Sideris Commonswiki, "*Dendrograma*", *2005.* https://es.wikipedia.org/wiki/Dendrograma (accedido el 18 de julio de 2024)

[16] María Morales, Carmelo Rodríguez, Antonio Salmerón "*Red bayesiana del ejemplo robo o terremoto*", *2007* https://www.researchgate.net/figure/Figura-2-Red-bayesiana-del-ejemplo-robo-o-terremoto_fig1_236985812 (accedido el 20 de julio de 2024)

[17] Towards Data Science, Tony Yiu, "*Understanding Random Forest"*, 2019. https://towardsdatascience.com/understanding-random-forest (accedido el 14 de noviembre de 2024)

[18] DataCamp, Bharath K, *"Introduction to Deep Neural Networks"*, 2024. https://www.datacamp.com/es/tutorial/introduction-to-deep-neural-networks (accedido el 15 de noviembre de 2024)

[19] InteractiveChaos, *"Estructura de una red neuronal (Tutorial de Machine Learning)"*. https://interactivechaos.com/es/manual/tutorial-de-machine-learning/estructura-de-una-red-neuronal (accedido el 15 de noviembre de 2024)

[20] DataCamp, Sejal Jaiswal, *"Multilayer Perceptrons in Machine Learning: A Comprehensive Guide"*, 2024. https://www.datacamp.com/tutorial/multilayer-perceptrons-in-machine-learning (accedido el 15 de noviembre de 2024)

[21] Edureka, Anirudh Rao, *"Recurrent Neural Networks (RNN) Tutorial | Analyzing Sequential Data Using TensorFlow in Python"*, 2023. https://www.edureka.co/blog/recurrent-neural-networks (accedido el 16 de noviembre de 2024)

[22] Wikipedia, *"Recurrent Neural Network"*, 2023. https://en.wikipedia.org/wiki/Recurrent_neural_network (accedido el 16 de noviembre de 2024)

[23] IBM, "*Understanding LSTMs and How They Work*". https://www.ibm.com/cloud/learn/lstm (accedido el 16 de noviembre de 2024)

[24] Wikipedia, *"Long Short-Term Memory"*. https://en.wikipedia.org/wiki/Long_short-term_memory (accedido el 16 de noviembre de 2024)

[25] Vaswani, Ashish; Shazeer, Noam; Parmar, Niki; Uszkoreit, Jakob; Jones, Llion; Gomez, Aidan N; Kaiser, Łukasz; Polosukhin, Illia (December 2017). "Attention is All you Need". In I. Guyon and U. Von Luxburg and S. Bengio and H. Wallach and R. Fergus and S. Vishwanathan and R. Garnett (ed.). 31st Conference on Neural

Information Processing Systems (NIPS). Advances in Neural Information Processing Systems. Vol. 30. Curran Associates, Inc. arXiv:1706.03762

[26] Towards Data Science, Amanda Iglesias Moreno *"Data Normalization with Pandas and Scikit-Learn"*, 2020. https://towardsdatascience.com/data-normalization (accedido el 17 de noviembre de 2024)

[27] Wikipedia, "*Activation Function*", 2024 https://en.wikipedia.org/wiki/Activation_function (accedido el 17 de noviembre de 2024)

[28] DataCamp, Bex Tuychiev, *"Stochastic Gradient Descent in Python: A Complete Guide for ML Optimization"*, 2024 https://www.datacamp.com/community/tutorials/stochastic-gradient-descent (accedido el 17 de noviembre de 2024)

[29] David Morán Montero "*Análisis de datos de telemetría de Fórmula 1 con técnicas de Deep Learning",* 2022.

[30] Ana Urquidi Castro "*Aplicación para la predicción y la optimización de las posiciones de las competiciones de Fórmula 1 mediante algoritmos de Machine Learning*", 2022.

[31] Python Software Foundation, *Python.org.* https://www.python.org/ (accedido el 19 noviembre de 2024)

[32] *Logo oficial de Python*, 2008. https://es.m.wikipedia.org/wiki/Archivo:Python-logo-notext.svg (accedido el 19 de noviembre de 2019)

[33] Paradigma Digital, "*Projections of future traffic for major programming languages*". https://www.paradigmadigital.com/wp-content/uploads/2017/11/grafica2.png (accedido el 19 de noviembre de 2024)

[34] Google, Google Colaboratory. https://colab.google/ (accedido el 20 de noviembre de 2024)

[35] Project Jupyter, *"Jupyter.org"*. https://jupyter.org/ (accedido el 20 de noviembre de 2024)

[36] Visual Studio Code. https://code.visualstudio.com/ (accedido el 20 de noviembre de 2024).

[37] Wikipedia, "*Logo oficial de Visual Studio Code*" https://es.wikipedia.org/wiki/Visual_Studio_Code (accedido el 20 de noviembre del 2024)

[38] Autor desconocido, *FastF1*. https://docs.fastf1.dev/ (accedido el 20 de noviembre de 2024)

[39] Phyton Data Analysis Library, Pandas https://pandas.pydata.org/ (accedido el 21 de noviembre de 2024)

[40] Wikipedia, *Logo Oficial de Panda, 2019.* https://en.m.wikipedia.org/wiki/File:Pandas_logo.svg (accedido el 21 de noviembre de 2024)

[41] Matplotlib (página oficial) https://matplotlib.org/ (accedido el 21 de noviembre de 2024)

[42] NumPy (página oficial) https://numpy.org/ (accedido el 21 de noviembre de 2024)

[43] Pytorch (página oficial) https://pytorch.org/ (accedido el 21 de noviembre de 2024)

[44] Wikipedia, *Logo de Pytorch*, 2020. https://es.m.wikipedia.org/wiki/Archivo:PyTorch_logo_black.svg (accedido el 21 noviembre de 2024)

[45] FastAPI https://fastapi.tiangolo.com/ (accedido el 21 de noviembre de 2024)

[46] Scikit-learn https://scikit-learn.org/stable/ (accedido el 21 de noviembre de 2024)

[47] Wikipedia, *Logo oficial de Scikit-learn, 2012.* https://en.m.wikipedia.org/wiki/File:Scikit_learn_logo_small.svg (accedido el 21 de noviembre de 2024)