MODELOS PREDICTIVOS DE RESULTADOS DEPORTIVOS: APLICACIÓN DE MACHINE LEARNING EN PLATAFORMAS DE APUESTAS

MODALIDAD DEL TFG: CONVENCIONAL

CONVOCATORIA: ORDINARIA

ALUMNA: ANA ISABEL GONZÁLEZ SAHAGÚN

TUTORA: ICÍAR CIVANTOS GÓMEZ

GRADO: INGENIERÍA DEL SOFTWARE

AGRADECIMIENTOS

Me gustaría agradecer a mis padres por brindarme la oportunidad de estudiar esta carrera y acompañarme en todo momento.

También quiero agradecer a mis profesores por su apoyo constante a lo largo de estos cinco años.

CONTENIDO

AGRADECIMIENTOS 2

RESUMEN 6

ABSTRACT 6

1. INTRODUCCIÓN 7

1.1 Motivación y Contexto 7

1.2 Planteamiento del Problema 8

1.2.1 Definición de Éxito de los Equipos 8

1.2.2 Selección del Modelo y Limitaciones 10

1.2.3 Necesidad de Definir Métricas Claras para Evaluar el Modelo 10

1.2.4 Tema poco desarrollado 11

1.3 Objetivos del Trabajo 11

2. ESTADO DE LA CUESTIÓN 12

2.1 Plataformas de Apuestas Deportivas 12

2.1.1 Funcionamiento de las Apuestas Deportivas 12

2.1.2 Datos Estadísticos del Mercado: Impacto Económico y Social 14

2.2 El Fútbol y las Apuestas 14

2.2.1 Tipos de Apuestas en el Fútbol 15

2.2.2 Impacto Económico de las Apuestas en el Fútbol 17

2.3 Marco Teórico del Trabajo 17

2.3.1 Introducción al Machine Learning 17

2.3.2 Machine Learning: Conceptos Básicos 20

2.3.3 Machine Learning: Evaluación del Modelo 25

2.3.4 Machine Learning: Modelos Lineales 28

2.3.5 Machine Learning: Modelos no Lineales 30

2.3.6 Machine Learning: Redes Neuronales 34

2.4 Trabajos Relacionados 37

3. ASPECTOS METODOLÓGICOS 39

3.1 Metodología 39
3.1.1 Planificación 39
3.1.2 Organización y seguimiento del trabajo 40
3.1.3 Adaptaciones y ajustes en la metodología 40
3.2 Tecnologías Empleadas 41
3.2.1 Lenguajes y entornos de desarrollo 41
3.2.2 Bibliotecas y Frameworks 41
3.2.3 Github 41
3.2.4 MLflow 42
4. DESARROLLO DEL TRABAJO 44
4.1 Adquisición, Análisis y Procesamiento de Datos 44
4.1.1 Descripción y Preproceso de los Datos 45
4.1.2 Análisis Exploratorio de los Datos 48
4.1.3 Generación de Datasets por Equipos 51
4.1.4 Análisis de Datasets por Equipos 54
4.2 Desarrollo del modelo 58
4.2.1 Introducción a los Modelos 58
4.2.2 Regresión Logística 60
4.2.3 KNN: K – Nearest Neighbors 63
4.2.4 Random Forest 66
4.2.4 Gradient Boosting y XGBoost 68
4.2.4 Redes Neuronales 71
4.2.5 Mejores Resultados 72
4.2.6 Conclusión General de los Resultados 73
5. CONCLUSIONES 78
5.1 Objetivos Planteados y Grado de Cumplimiento 78
5.2 Limitaciones 79
5.3 Discusión y Análisis Crítico 80

5.4 Propuestas de Trabajos Futuros 81
5.5 Conclusión Personal 82
6. REFERENCIAS 83
6.1 Bibliografía 83
6.2 Índices de Imágenes y Figuras 85
6.2.1 Imágenes 85
6.2.2 Figuras 87
ANEXOS 88

RESUMEN

El *Machine Learning* se ha convertido en una herramienta fundamental para el análisis de datos complejos. En el ámbito deportivo, su aplicación en la predicción de resultados ofrece oportunidades interesantes, pero también desafíos importantes. Las casas de apuestas, con su alto impacto económico, añaden además una dimensión ética relevante al uso de este tipo de modelos.

Este estudio explora distintos enfoques de aprendizaje automático para predecir el resultado de partidos de fútbol (victoria, empate o derrota), con datos históricos de varios equipos de La Liga Española. Aunque la naturaleza impredecible del deporte limita la precisión alcanzable, los resultados muestran que una buena selección de variables y técnicas de validación puede mejorar notablemente la capacidad de los modelos para identificar los distintos resultados. El objetivo de este trabajo es evaluar hasta qué punto el *Machine Learning* puede enfrentarse a escenarios con un alto grado de aleatoriedad, como es el caso del fútbol profesional.

ABSTRACT

Title: Predictive Models for Sports Outcomes: Application of Machine Learning in Betting Platforms

Machine Learning has become a key tool for analyzing complex data. In the sports domain, its application to match outcome prediction offers promising opportunities but also presents significant challenges. Betting markets, due to their strong economic impact, introduce an additional ethical dimension when using predictive models in this context. This study explores different Machine Learning approaches to predict the outcome of football matches (win, draw, or loss), using historical data from several teams in Spain's La Liga. While the unpredictable nature of the sport limits the achievable accuracy, the results show that a careful selection of variables and validation strategies can significantly improve model performance. The aim of this work is to assess how effectively Machine Learning can deal with highly random environments such as professional football.

1. INTRODUCCIÓN

1.1 Motivación y Contexto

El deporte, además de ser una gran fuente de entretenimiento, mueve cifras económicas notables en todo el mundo. Las competiciones deportivas atraen a seguidores de todas las edades y países, generando ingresos millonarios a través de la venta de entradas, derechos de transmisión, patrocinio y publicidad. El deporte se ha convertido en un factor decisivo para el crecimiento económico y el desarrollo social de muchos países.

Dentro de este ámbito, el fútbol destaca como el deporte con mayor número de aficionados, impulsando la creación de empleo y fomentando la actividad comercial a su alrededor. En 2023, la famosa empresa consultora KPMG realizó un estudio sobre el impacto socioeconómico del fútbol profesional en España. Según este informe, el fútbol profesional alcanzó un impacto económico de 18.350 millones de euros durante la temporada 2021/2022, equivalente al 1,44% del PIB nacional [1]. Además, las apuestas deportivas en línea rebasaron los 2.950 millones de euros en ese mismo periodo.

Las plataformas de apuestas se enfocan en anticipar de la forma más precisa posible el resultado de estos encuentros deportivos. Esto se debe a que un cálculo correcto de estas predicciones afecta de manera directa en sus ganancias y en las decisiones de quienes apuestan. Dado el impacto económico de las plataformas de apuestas, mejorar la precisión en la predicción de resultados puede tener un efecto significativo en la rentabilidad de estas plataformas y en la estrategia de los apostadores. Esto ha despertado un creciente interés por parte de la comunidad científica y tecnológica en el desarrollo de modelos predictivos más precisos, especialmente mediante técnicas de Inteligencia Artificial y Machine Learning.

Sin embargo, es importante destacar la dificultad inherente de este problema. El fútbol es un deporte con una fuerte naturaleza estocástica, donde intervienen factores difíciles de modelar como el estado físico de los jugadores, decisiones arbitrales o lesiones imprevistas. Existen numerosos estudios como "*Luck is Hard to Beat: The Difficulty of Sports Prediction*" que destacan como la mezcla de habilidad y azar en un encuentro deportivo hace que ningún modelo sea capaz de asegurar un resultado final con total exactitud [2].

Precisamente, este componente impredecible del deporte, y en particular del fútbol, lo convierte en un ámbito interesante para el estudio de Inteligencia Artificial y Machine Learning. Actualmente, la IA es un campo en pleno auge que ha experimentado un crecimiento acelerado en los últimos años. Su aplicación en el ámbito deportivo ha permitido optimizar estrategias tácticas, mejorar el rendimiento de los jugadores y prevenir lesiones mediante el análisis de

patrones físicos y niveles de fatiga. No obstante, la predicción de resultados sigue siendo un área poco explorada debido a la gran cantidad de elementos inciertos que influyen en el desarrollo de un partido.

En este contexto, el presente estudio tiene como objetivo explorar hasta qué punto las técnicas de Machine Learning pueden aplicarse eficazmente en la predicción de situaciones con un alto componente de aleatoriedad, como ocurre en el deporte, y en particular en el fútbol.

1.2 Planteamiento del Problema

Predecir resultados deportivos es un reto complejo que requiere la capacidad de analizar múltiples elementos que pueden influir en el desarrollo de un partido. Cada partido y cada jugada está influida por una combinación infinita de factores, como la forma física y mental de los jugadores, las estrategias empleadas, las condiciones climatológicas externas, y el azar. Esta tarea representa un gran desafío técnico, pero también una oportunidad con impacto significativo. ¿Es posible que un modelo de Machine Learning sea capaz de detectar patrones ocultos y convertir esa aleatoriedad en algo más predecible? Para desarrollar el problema, se ha utilizado un conjunto de datos históricos de partidos de la liga española desde 2003. A continuación, se presentan los principales desafíos que enfrenta este problema.

1.2.1 Definición de Éxito de los Equipos

Determinar qué significa el éxito en el fútbol es clave para construir modelos de predicción precisos. La victoria de un encuentro es la medición inmediata más directa para evaluar el éxito de un equipo. Sin embargo, la definición de éxito toma matices específicos según el interés del estudio. Por ejemplo, para medir el rendimiento de un equipo se pueden considerar factores como la clasificación final en una liga, la diferencia de goles, el porcentaje de victorias de una temporada, y otros indicadores relacionados con el desempeño. Existe la necesidad de realizar un análisis previo para determinar cuáles de estas variables resultan más útiles a la hora de desarrollar modelos de predicción e interpretar los resultados. A continuación se presentan dos métricas utilizadas en diferentes estudios para analizar el rendimiento de los equipos.

Soccer Power Index (SPI):

El SPI es un sistema de calificación desarrollado por FiveThirtyEight que estima la calidad de los equipos de fútbol a nivel mundial[1]. Este índice combina datos históricos con un modelo predictivo que evalúa el rendimiento de los modelos en función de su capacidad ofensiva y defensiva. Se basa en los siguientes componentes:

1. Calificación de ataque: Estima la cantidad de goles esperados que un equipo marcaría contra un oponente promedio.
2. Calificación de defensa: Estima la cantidad de goles que un equipo recibiría contra un oponente promedio.
3. Resultados recientes: Se ponderan más los partidos recientes, pero sin ignorar completamente los datos históricos.

Expected Goals (xG)

Es una de las métricas más influyentes en la analítica moderna del fútbol, utilizada en numerosos estudios [3]. En lugar de medir simplemente los coles marcados, el xG evalúa la calidad de cada oportunidad de gol basándose en múltiples factores:

1. Ubicación del disparo: La distancia y el ángulo con respecto a la portería influyen en la probabilidad de éxito
2. Tipo de pase previo: Dependiendo del pase, la probabilidad del gol puede cambiar
3. Tipo de disparo: Disparos con el pie, cabezazos o situaciones uno contra uno. Cada uno tiene probabilidades distintas de éxito.
4. Situación de juego: Los tiros en jugadas a balón parado, penaltis, o en jugadas abiertas tienen distintos valores de xG.

Un xG alto significa que el equipo ha generado oportunidades de alta calidad, incluso si no ha marcado goles. Estudios como *Beyond Expected Goals* del MIT han demostrado que el xG es un predictor más confiable del rendimiento ofensivo futuro en comparación con los goles marcados reales [4]. Sin embargo, la tarea de recopilación de datos para calcular el xG es mucho más compleja, ya que requiere un alto nivel de precisión en la medición de variables como la ubicación del disparo, la velocidad del balón y la presión defensiva en cada jugada.

---

[1] ESPN staff (2014, 11 junio). Soccer Power Index explained. https://www.espn.com/

1.2.2 Selección del Modelo y Limitaciones

Existen diversos enfoques utilizados en la predicción de resultados deportivos, cada uno con sus ventajas y sus limitaciones. Modelos como las Redes Neuronales Recurrentes han mostrado resultados prometedores en la predicción de partidos. Son ampliamente utilizados para calcular probabilidades de victoria en encuentros de fútbol debido a su capacidad para manejar grandes volúmenes de datos y detectar patrones complejos.

A pesar de los avances recientes, los modelos de predicción siguen enfrentando varias limitaciones. Estudios muy recientes siguen destacando estas dificultades [5], entre ellas:

1. Calidad de los datos: La precisión de las predicciones depende en gran medida de la calidad y la cantidad de datos disponibles.
2. Sobreajuste a patrones históricos: Muchos modelos tienden a ajustarse demasiado a los datos pasados, lo que reduce su capacidad de generalizar en situaciones nuevas o imprevistas.
3. Imprevisibilidad inherente de los resultados deportivos: Aspectos como lesiones inesperadas o decisiones arbitrales siguen siendo difíciles de predecir con precisión y pueden influir significativamente en el resultado de un partido.

1.2.3 Necesidad de Definir Métricas Claras para Evaluar el Modelo

Para medir la efectividad de los modelos de predicción es fundamental definir métricas de evaluación adecuadas. Un desafío clave es que ninguna métrica es completamente representativa por sí sola. La combinación de varias medidas permite una validación más rigurosa del modelo. Algunas de las más utilizadas incluyen:

- Accuracy: Porcentaje de predicciones correctas. Sin embargo, no siempre es una métrica adecuada en deportes donde los empates y la variabilidad afectan los resultados.
- Log Loss: Evalúa la probabilidad asignada a cada resultado, penalizando predicciones con alta confianza pero incorrectas.
- Área bajo la curva ROC (AUC-ROC): Mide la capacidad del modelo para distinguir entre diferentes categorías (victoria, empate, derrota).
- F1-Score: Es la media armónica entre precisión (precision) y exhaustividad (recall). Resulta especialmente útil cuando existe un desequilibrio entre clase.

1.2.4 Tema poco desarrollado

A pesar del creciente interés en el uso de Machine Learning en la analítica deportiva, la predicción de resultados deportivos sigue siendo un área con muchas limitaciones. Hasta la fecha, no existe un modelo ampliamente aceptado que haya demostrado predecir con precisión los resultados de fútbol. Muchos estudios presentan buenos resultados en ligas específicas o temporadas concretas, pero fallan al generalizar en otros contextos. Además, no existe un consenso sobre qué modelo o conjunto de técnicas es el más adecuado para este tipo de predicciones.

1.3 Objetivos del Trabajo

Este proyecto tiene como objetivo general explorar y evaluar la capacidad de los modelos de Machine Learning para realizar predicciones en un entorno caracterizado por su alta imprevisibilidad, como un partido de fútbol. En particular, se busca determinar si la inteligencia artificial es capaz de predecir una situación de gran aleatoriedad, y hasta qué punto puede reducir la incertidumbre en los resultados. Para lograrlo, se establecen los siguientes objetivos específicos, que guiarán el desarrollo del trabajo:

- **Mejorar la eficiencia de los modelos:** Trabajar con técnicas para ordenar y extraer características clave, facilitando criterios de clasificación claros.
- **Desarrollo y evaluación de modelos**: Implementar diferentes modelos de Machine Learning, ajustarlos y evaluar su rendimiento en función de métricas previamente definidas.
- **Predicción de resultados multiclase** Determinar si un equipo ganará, perderá, o empatará un partido.
- **Comparar la efectividad de los distintos modelos propuestos:** Ofrecer una comparativa del rendimiento de los distintos modelos propuestos para identificar cuál ofrece la mejor precisión en la predicción de resultados.
- **Definir y aplicar métricas de evaluación**: Establecer métricas claras para medir la efectividad de los modelos y analizar sus resultados.
- **Examinar limitaciones y efectividad**: Evaluar las limitaciones de los modelos propuestos, así como su efectividad para reducir la aleatoriedad inherente al ámbito deportivo.

2. ESTADO DE LA CUESTIÓN

2.1 Plataformas de Apuestas Deportivas

Las apuestas deportivas han evolucionado significativamente a lo largo del tiempo, desde prácticas informales en la antigüedad hasta convertirse en una industria multimillonaria. Su desarrollo ha estado marcado por varios hitos importantes, como la regulación de las apuestas en Inglaterra durante el siglo XVIII.[2] Más recientemente, la transformación digital impulsada por Internet ha facilitado la aparición de plataformas en línea, que permiten realizar apuestas en tiempo real. En este apartado se ofrecerá una breve contextualización del mundo de las apuestas deportivas.

2.1.1 Funcionamiento de las Apuestas Deportivas

Las casas de apuestas funcionan estableciendo cuotas que reflejan la probabilidad de que ocurra un determinado resultado en un evento deportivo. Las cuotas se calculan utilizando análisis estadísticos y modelos matemáticos que consideran diferentes factores.

Las casas de apuestas aplican un margen de beneficio llamado "*vig*" o "*juice*", incorporado en las cuotas ofrecidas a los apostadores para asegurar ganancias a largo plazo. Además, se ajustan las cuotas en función del volumen de apuestas recibidas para equilibrar su exposición y garantizar beneficios independientemente del resultado. Este margen se obtiene ofreciendo cuotas ligeramente inferiores a las probabilidades reales del evento, lo que permite que la suma de las probabilidades implícitas supere el 100%. Además, Las casas incorporan sesgos de mercado, ajustes de liquidez, y consideraciones sobre el volumen total apostado para equilibrar su exposición al riesgo y minimizar pérdidas. Este ajuste se intensifica en apuestas en vivo, donde las cuotas se actualizan en directo según transcurre el evento.

Tipos de Casas de Apuestas

Existen dos tipos principales de casas de apuestas:

---

[2] La evolución histórica de las apuestas deportivas: de los juegos antiguos a la era digital. (2023, 21 junio). Fox Sports. https://www.foxsports.com.

- **Casas de apuestas de contrapartida**: Son las más comunes y operan estableciendo cuotas y aceptando apuestas directamente contra los apostantes. Ejemplos de este tipo incluyen Bet365, William Hill y Bwin.
- **Casas de apuestas de intercambio**: Permiten que los apostantes jueguen entre sí en lugar de hacerlo contra la casa. En este modelo, los usuarios pueden fijar sus propias cuotas y aceptar apuestas de otros jugadores. La casa actúa como intermediaria, obteniendo un beneficio a través de una comisión sobre las ganancias de los jugadores. Un ejemplo destacado es Betfair.

Ajustes de Cuotas y Margen de Beneficio: Sesgos e Ineficiencias

Para protegerse de posibles pérdidas, las casas de apuestas incluyen un margen de seguridad en las cuotas. Este margen está basado en la probabilidad estadística de los resultados y en el comportamiento de los apostantes. Existen varios estudios que investigan los sesgos y las ineficiencias de las casas de apuestas [6]. Entre los más importantes se encuentran:

- Longshot Bias: Se refiere a la tendencia de sobrevaloración de las apuestas a resultados poco probables, implicando que la probabilidad real de ganar es subestimada en las cuotas [7].
- Home-Field Advantage Misestimation: Error al estimar la ventaja del equipo local. Tradicionalmente, se considera que el equipo de casa tiene una mayor probabilidad de ganar, lo que se refleja en cuotas más bajas. Pero existen situaciones como la ausencia del público que pueden modificar este efecto.
- Incorporación Tardía de Información: Ocurre cuando las casas de apuestas actualizan de forma lenta las cuotas ante la aparición de nueva información.

Modelo de rentabilidad

La rentabilidad de una casa de apuestas se basa en la diferencia entre las cantidades apostadas y los pagos realizados a los ganadores. El modelo de negocio de las casas de apuestas no se basa en acertar los resultados de los eventos, sino de asegurarse ganancias a largo plazo ajustando las cuotas y controlando el riesgo financiero [8]. El margen de beneficio se puede expresar con la formula:

$$Beneficios = (Cuota\ Ganadora * Cantidad\ Apostada) - Cantidad\ Apostada$$

Si el resultado es desfavorable para la casa, su perdida se limita a la cantidad apostada

$$Pérdidas\ =\ Cantidad\ Apostada$$

2.1.2 Datos Estadísticos del Mercado: Impacto Económico y Social

Estudios recientes estiman que el mercado global de apuestas deportivas genera más de 90 millones de dólares anuales. Esta cifra tiene tendencia a seguir aumentando debido a la legalización en varios países y acceso facilitado por plataformas digitales. En regiones como Europa y América del Norte, las apuestas deportivas son un gran porcentaje la industria del entretenimiento y juegos de azar. Se estima que el mercado de las apuestas deportivas previsto para 2030 sea de 608.410 millones USD. [3]

Por otra parte, la accesibilidad de las apuestas en línea ha aumentado el número de casos de adicción al juego, especialmente entre los más jóvenes. Estudios estiman que alrededor de 80 millones de adultos en todo el mundo sufren de adicción al juego.[4] En países como Brasil se ha observado una preocupación creciente por el gasto excesivo en apuestas en línea. Informes indican que los brasileños gastan más de 3.200 millones de euros mensualmente en apuestas, representando el 20% de la masa salarial del país.[5]

2.2 El Fútbol y las Apuestas

El fútbol es el deporte más popular en el ámbito de las apuestas deportivas. Este deporte tiene una gran base global de aficionados y una constante programación de partidos diarios a lo largo del año, representando más del 86% de las actividades de apuestas deportivas en algunos países.[6]

---

[3] Fernández, R. (2025, 13 febrero). Las apuestas y los juegos de azar en el mundo: Datos estadísticos. https://es.statista.com/

[4] Mouzo, J. (2024, 24 octubre). La amenaza de llevar un casino en el bolsillo: 80 millones de adultos sufren adicción al juego. El País. https://elpais.com

[5] Zuppello, M. (2024, 30 septiembre). Adicción a las apuestas online en Brasil. Infobae. https://www.infobae.com

[6] (2024, abril) El Pilón. Los favoritos de los aficionados: Los deportes más populares para apostar. https://elpilon.com.co

2.2.1 Tipos de Apuestas en el Fútbol

Las apuestas deportivas en el fútbol han evolucionado en las últimas décadas con la digitalización y legalización de plataformas de juego en línea. Aunque existen múltiples modalidades de apuestas, este trabajo se centrará en la siguiente:

1-X-2: Apuesta a qué equipo ganará el partido, o si habrá un empate.

Cada opción tiene una cuota asociada, que representa el pago potencial por cada unidad apostada. Estas cuotas pueden variar entre casas de apuestas y fluctuar en función de diversos factores como estadísticas previas, alineaciones, lesiones o la cantidad de dinero apostado en cada opción.

Imagen 1 – Plataforma de resultados Flashscore: Apuestas 1X2

Si un apostador apuesta 1€ a la victoria del Atlético de Madrid en Bet365 con una cuota de 3.00, y acierta, recibiría 3.00€ en total: 2.00€ de ganancia neta más 1.00€ de la apuesta inicial.

- **Flecha arriba:** Indica que la cuota ha subido, lo que significa que la probabilidad percibida de ese resultado ha disminuido, o bien que hay menos apuestas en esa opción.
- **Flecha abajo**: indica que la cuota ha **bajado**, lo que sugiere que hay más dinero apostado en esa opción y, por lo tanto, se considera más probable.

Sin embargo, existen diversas modalidades de apuestas en el fútbol, cada una con características específicas que permiten diferentes estrategias de juego. Algunas de las más destacadas son:

1. Resultado exacto
2. Resultado al descanso y al final del partido
3. Doble oportunidad: Apostar a dos posibles resultados
4. Apuesta con reembolso en caso de empate.
5. Apuestas sobre goles totales.

6. Apuestas especiales: Apostar a diferentes aspectos del partido, como el número de tarjetas amarillas.
7. Scorecast: Acertar tanto el primer goleador como el resultado final.
8. Hándicap Asiático

El **hándicap asiático** consiste en asignar una ventaja o desventaja en el marcador a uno de los equipos antes del inicio del partido. Su principal característica es que elimina la posibilidad de empate, ya que en muchos casos se ofrecen líneas con fracciones (como -1.5 o -2.5), lo que asegura que siempre habrá un ganador en la apuesta.

- Hándicap -2.5 en Atlético de Madrid: El Atlético debe ganar por al menos 3 goles para que la apuesta sea ganadora.
- Hándicap -2: Si gana por exactamente 2 goles, se reembolsa la apuesta.
- Hándicap -1.75: Si gana por 2 goles, se gana la mitad de la apuesta y la otra mitad se devuelve.
- Flechas arriba: Indican menor probabilidad percibida.
- Flechas abajo: Indican mayor confianza en esa opción.

Imagen 2 – Plataforma de resultados Flashscore: Hándicap Asiático

Dado que tanto las cuotas 1X2 como las de hándicap asiático están disponibles antes del inicio del partido, se incluirán estas variables en los modelos de predicción. Otras modalidades, como el número de tarjetas, el primer goleador o el resultado exacto, no se han incluido por falta de datos o por no estar directamente relacionadas con el objetivo de predecir el desenlace del partido. Incluir las cuotas 1X2 y de hándicap como variables de entrada puede aportar información valiosa, ya que reflejan de forma implícita el conocimiento del mercado y las expectativas sobre el partido, lo que podría mejorar la precisión del modelo.

2.2.2 Impacto Económico de las Apuestas en el Fútbol

La Copa Mundial y la UEFA Champions League son los eventos futbolísticos más populares de la actualidad y ambos tienen un gran impacto económico en el mercado de apuestas. Durante estos torneos, se observa un aumento considerable en la actividad de apuestas, lo que genera fluctuaciones económicas notables. Por ejemplo, en España, las apuestas deportivas representan casi el 1% del PIB nacional. [1]

El fútbol no solo tiene un impacto en la industria del juego, sino que también influye en la economía general del país. La actividad económica generada por el futbol repercute en la generación de empleo, con más de 194.381 empleos a jornada completa, incluyendo puestos directos e indirectos relacionados con el deporte.[7]

2.3 Marco Teórico del Trabajo

El Machine Learning permite modelar relaciones complejas y no lineales entre múltiples variables, algo difícil de capturar con técnicas estadísticas tradicionales. En el caso del fútbol, donde intervienen factores impredecibles y datos heterogéneos, los modelos de Machine Learning pueden aprender patrones ocultos a partir de datos históricos y ajustarse a la alta variabilidad del juego, lo que los convierte en una herramienta especialmente útil en contextos con incertidumbre, como las apuestas deportivas.

En este apartado se presentan los conceptos teóricos fundamentales para el desarrollo del trabajo. Esta sección se basa en un conjunto de artículos introductorios que explican los fundamentos del Machine Learning. Solo se explican los conceptos básicos necesarios para comprender el desarrollo del trabajo.

2.3.1 Introducción al Machine Learning

El Machine Learning (ML) es una subrama de la Inteligencia Artificial que se centra en desarrollar algoritmos capaces de aprender y hacer predicciones a partir de datos, sin ser programados específicamente para ello. Algunas de las aplicaciones del Machine Learning son:

- Reconocimiento de voz para asistentes virtuales.

---

[7] Villar, G. (2023, 13 octubre). La cara B de la riqueza que genera el fútbol: el 43% del gasto de los aficionados va a las apuestas online. Relevo. https://www.relevo.com

- Traducción automática de textos en tiempo real.
- Finanzas y predicción de mercados.
- Diagnóstico de enfermedades a partir de imágenes médicas.

Funcionamiento

El funcionamiento del Machine Learning se basa en el desarrollo de modelos matemáticos que, a través de la experiencia y la optimización de funciones, pueden mejorar su desempeño en diversas tareas. Estos modelos matemáticos establecen una relación entre las variables de entrada (*features)* y las variables de salida (*targets),* con el objetivo de minimizar una función de error o pérdida. [10] Los modelos aprenden a partir de un conjunto de datos mediante un proceso iterativo de optimización donde se ajustan los parámetros internos para mejorar la precisión.

Matemáticamente, el aprendizaje de los modelos de Machine Learning se puede representar como la búsqueda de una función desconocida:

$$f: X \rightarrow Y$$

que mapea un conjunto de entradas X a una salida Y, con objetivo de minimizar (o maximizar) una función objetivo $L(Y, \hat{Y})$. Esta función objetivo cuantifica la discrepancia entre la predicción $\hat{Y}$ y el valor real Y. Dependiendo del tipo de aprendizaje, los modelos de Machine Learning se clasifican en tres tipos comunes:

1. Aprendizaje supervisado
2. Aprendizaje no supervisado
3. Aprendizaje por refuerzo

Aprendizaje supervisado

El aprendizaje supervisado se basa en entrenar modelos utilizando un conjunto de datos etiquetado, es decir, donde cada entrada tiene una salida conocida. Su objetivo es aprender una función de mapeo que relacione las entradas con las salidas correctas para poder hacer predicciones precisas sobre nuevos datos. [10]

En términos matemáticos, este tipo de aprendizaje se basa en la construcción de una función:

$$f: X \rightarrow Y$$

que aprende una relación entre un conjunto de características de la entrada X y sus respectivas salidas Y. El modelo se entrena utilizando un conjunto de datos etiquetado:

$$D = (x_i, y_i)_{i=1}^{n}$$

donde x_i representa un vector de características y cada y_i es la etiqueta correspondiente. El objetivo es minimizar la función de pérdida $L(Y, \hat{Y})$ que cuantifica la discrepancia entre la predicción del modelo $\hat{Y}$ y el valor real Y.

Este es el tipo de aprendizaje que se utilizará para desarrollar este trabajo, dado que tenemos datos históricos de partidos etiquetados con sus respectivos resultados. En los siguientes apartados se detallará con mayor profundidad el funcionamiento de la función de pérdida, los diferentes tipos existentes y su impacto en la precisión del modelo

Aprendizaje no supervisado

En el aprendizaje no supervisado, el modelo se entrena con datos que no están etiquetados, es decir, que no hay una salida específica asignada a cada entrada. En lugar de predecir valores específicos, los modelos buscan encontrar agrupaciones o patrones entre los datos. [10]

Matemáticamente, se basa en la optimización de una función de similitud o dispersión, en lugar de minimizar una función de error explícita.

Estos modelos se aplican en tareas como la detección de anomalías y segmentación de clientes, donde no es necesario contar con datos etiquetados. Sin embargo, dado que este proyecto se centra en aprendizaje supervisado, los modelos no supervisados no serán explicados en detalle más adelante.

Aprendizaje por refuerzo

El aprendizaje por refuerzo, tambien llamado *Reinforcement Learning* en inglés, se basa en conseguir que el agente interaccione con un entorno con el objetivo de aprender una política óptima $\pi(s)$ que maximice la recompensa acumulada a lo largo del tiempo. El agente aprende una función de valor $Q(s, a)$ que estima la recompensa esperada al tomar una acción a en un estado s. En términos generales, el agente mejora su toma de decisiones a través de prueba y error, donde recibe retroalimentación en forma de recompensas o penalizaciones para ajustar su comportamiento y maximizar el retorno a largo plazo. [10]

Esta técnica de Machine Learning podría resultar interesante para la automatización de estrategias de apuestas deportivas en plataformas en línea. En este caso se entrenaría al agente para decidir cuándo apostar, por qué equipo apostar y cuánto apostar, basándose en

probabilidades calculadas a partir de datos históricos, con el objetivo de maximizar las ganancias. Sin embargo este tipo de modelo añade una complejidad adicional significativa al estudio, por lo que no será explicado más adelante.

2.3.2 Machine Learning: Conceptos Básicos

A continuación se presentan los conceptos básicos necesarios para comprender el funcionamiento de los modelos con los que se van a estar trabajando en este TFG.

Función de Pérdida

El objetivo de la función de pérdida es proporcionar una métrica cuantificable para que el algoritmo de optimización pueda minimizar el error y mejorar la precisión del modelo. Esta función mide la diferencia entre la predicción del modelo $\hat{Y}$ y el valor real Y. [23]

Sea un conjunto de entrenamiento etiquetado $D = (x_i, y_i)_{i=1}^{n}$, donde x_i representa un vector de características y cada y_i es la salida real. El modelo $f(x;\theta)$ genera una predicción $\hat{Y}$, y la función de pérdida se define como:

$$L(Y, \hat{Y}) = \frac{1}{N}\sum_{i=1}^{N} l(y_i - \hat{y}_i)^2$$

donde $l(y_i, \hat{y})^2$ mide el error de la predicción en una muestra individual.

La elección de la función de pérdida determina como el modelo ajusta los parámetros para minimizar el error. Dependiendo del tipo de modelo, se utilizan diferentes funciones de pérdida.

Para modelos de **clasificación** (predicción de etiquetas discretas) una de las funciones de pérdida más utilizadas es la **Entropía Cruzada (Cross-Entropy Loss)**, que busca maximizar la probabilidad asignada a la clase correcta.

- Clasificación binaria: dos etiquetas (o clases) posibles

$$L(Y, \hat{Y}) = -\sum_{i=1}^{N} y_i \log(\hat{y_i})$$

- Clasificación multiclase: varias etiquetas (o clases) posibles

$$L(Y, \hat{Y}) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} \log(\widehat{y_{i,j}})$$

Donde:

y_i es la verdadera clase

$\hat{y}_i$ es la probabilidad predicha por el modelo para la clase correcta

N es el número total de ejemplos en el conjunto de datos

C es el número de clases

Algoritmos de Optimización

Los algoritmos de optimización son métodos numéricos utilizados para encontrar los valores de los parámetros del modelo que minimizan las funciones de pérdida. En Machine Learning, estos algoritmos ajustan los pesos del modelo a lo largo de las iteraciones del entrenamiento. La optimización eficiente de la función de pérdida es clave para garantizar un entrenamiento estable y efectivo de los modelos. [23]

Descenso de Gradiente: El Descenso de Gradiente (Gradient Descent, GD) es el algoritmo más utilizado para optimizar los modelos de Machine Learning. La actualización de los parámetros se realiza con la siguiente regla:

$$\theta = \theta - \alpha \nabla L(\theta)$$

Donde

θ representa los parámetros del modelo

α es la tasa de aprendizaje (Learning Rate)

$\nabla L(\theta)$ es el gradiente de la función de pérdida con respecto a θ

Tasa de aprendizaje: La tasa de aprendizaje (α) es un hiperparámetro en los algoritmos de optimización que controla la magnitud del ajuste de los parámetros del modelo por cada iteración del entrenamiento.

- Valores demasiado grandes pueden causar inestabilidad al modelo
- Valores demasiado pequeños pueden ralentizar la convergencia del modelo

Variantes del Descenso de Gradiente: Existen diferentes variantes del algoritmo de optimización del Descenso de Gradiente para mejorar la eficiencia del modelo

- Descenso de Gradiente Estándar (Batch Gradient Descent): Usa todo el conjunto de datos en cada iteración para calcular el gradiente

- Descenso de Gradiente Estocástico (SGD - Stochastic Gradient Descent): Actualiza los parámetros después de cada muestra
- Adam (Adaptive Moment Estimation): Combina SGD con ajustes adaptativos de la tasa de aprendizaje.

Los algoritmos de optimización son fundamentales en modelos como la regresión logística, redes neuronales y técnicas de boosting, donde se ajustan los parámetros para minimizar una función de pérdida. Sin embargo, no se aplican directamente en modelos como Random Forest o KNN, que no requieren entrenamiento iterativo basado en optimización.

Overfitting y underfitting

Uno de los objetivos más importantes de los modelos de Machine Learning es la capacidad de generalizar bien a datos no vistos. En este contexto, existen dos problemas comunes:

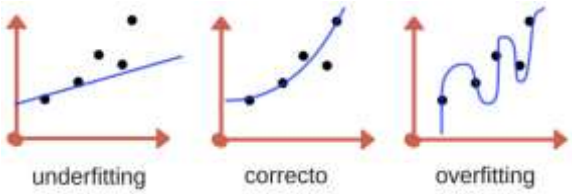

Imagen 3– Representación gráfica de los conceptos de overfitting y underfitting

Overfitting (Sobreajuste): Ocurre cuando el modelo se ajusta demasiado bien a los datos de entrenamiento, incluyendo ruido y patrones irrelevantes. Como resultado, tiene un bajo error en el entrenamiento pero un alto error en datos no vistos. [11] En el contexto del fútbol, donde muchos factores impredecibles afectan el resultado, es fundamental evitar el sobreajuste para que el modelo pueda generalizar correctamente a nuevos partidos.

Soluciones:

- Regularización L1: Agrega una penalización a los parámetros del modelo
- Regularización L2: Reduce la magnitud de los coeficientes del modelo para mejorar la generalización
- Dropout: En redes neuronales, desactiva neuronas aleatoriamente en cada iteración.
- Aumento de datos (Data Augmentation): Genera más datos a partir de los existentes.
- Validación cruzada: Divide los datos en múltiples conjuntos de entrenamiento y prueba.

Underfitting (Subajuste): Sucede cuando el modelo es demasiado simple para capturar la estructura de los datos resultando en alto error tanto en el entrenamiento como en la validación.

Soluciones:

- Aumentar la complejidad del modelo (ej., más capas en redes neuronales).
- Agregar más características relevantes en los datos de entrada.
- Entrenar por más épocas para permitir una mejor adaptación a los datos.

Hiperparámetros

Los hiperparámetros son valores que se deben definir antes de la optimización del modelo para configurar su entrenamiento.

Ejemplos:

- Tasa de aprendizaje (α): Controla cuánto se actualizan los parámetros en cada iteración.
- Número de árboles en Random Forest: Influye en la estabilidad y precisión del modelo.
- Número de capas y neuronas en Redes Neuronales: Afecta la capacidad de representación del modelo.

Los hiperparámetros pueden ajustarse manual o automáticamente mediante técnicas de optimización. En la búsqueda automatizada, se define un rango de valores posibles para cada hiperparámetro y el modelo realiza múltiples entrenamientos con distintas combinaciones. Finalmente, selecciona la configuración óptima basada en el desempeño obtenido según una métrica de evaluación predefinida. [24]

- Grid Search: Explora de forma exhaustiva todas las combinaciones posibles dentro de un espacio definido de hiperparámetros. Es fácil de implementar pero computacionalmente costoso, especialmente cuando el número de parámetros o el rango de valores es grande.
- Random Search: Selecciona combinaciones aleatorias dentro del mismo espacio. Aunque no garantiza explorar todas las combinaciones, suele ser más eficiente que Grid Search y permite descubrir buenas configuraciones con menos evaluaciones.
- Optuna: Es una herramienta de optimización bayesiana que ajusta automáticamente la búsqueda de hiperparámetros utilizando diferentes técnicas para detener pruebas poco prometedoras. Es más eficiente y escalable en comparación con Grid y Random Search.

Reducción de dimensionalidad

En Machine Learning, manejar conjuntos de datos con muchas variables puede generar redundancia y afectar la eficiencia del modelo. La reducción de dimensionalidad es un proceso que busca representar información con menos variables, eliminando las que son redundantes o tienen poca influencia en la predicción. [23] A continuación se presentan las técnicas más comunes.

Selección de características

Consiste en elegir las variables más relevantes y descartar las que aportan poca información o son redundantes. Esta técnica mantiene las variables originales y es útil cuando algunas características son irrelevantes para el problema. Se divide en:

- Métodos de filtro: Aplican medidas estadísticas para evaluar la relevancia de cada variable.
- Métodos de envoltura: Evalúan distintos subconjuntos de variables con un modelo predictivo.
- Métodos basados en árboles de decisión: Identifican la importancia de cada variable en la predicción.

Análisis de componentes principales (PCA)

PCA es una técnica de reducción de dimensionalidad que transforma las variables originales en un nuevo conjunto de variables no correlacionadas. Estas nuevas variables se llaman **componentes principales**.

Matemáticamente, PCA busca encontrar una base ortogonal de menor dimensión en la que la varianza de los datos se preserve en la mayor medida posible. El proceso implica:

1. Estandarización de las variables para que tengan media cero y varianza unitaria.
2. Cálculo de la matriz de covarianza para identificar relaciones entre variables.
3. Descomposición en valores propios, extrayendo los vectores propios que representan las direcciones de mayor varianza.
4. Selección de los componentes principales, eligiendo los que explican la mayor parte de la varianza.

A diferencia de la selección de características, PCA no elige variables específicas, sino que las reestructura, lo que puede dificultar la interpretación directa de los datos. En este trabajo se analizará el impacto de las diferentes técnicas para reducir la dimensionalidad de los datos. Se

realizarán comparaciones entre modelos entrenados con y sin la aplicación de estas técnicas, con el objetivo de evaluar su efecto en la precisión, eficiencia y capacidad de generalización de los modelos de predicción.

2.3.3 Machine Learning: Evaluación del Modelo

Para evaluar correctamente el rendimiento de un modelo de Machine Learning, es necesario dividir el conjunto de datos en al menos dos subconjuntos:

- **Entrenamiento (Training set):** utilizado para ajustar los parámetros del modelo.
- **Prueba (Test set):** utilizado para evaluar el rendimiento final del modelo sobre datos no vistos.

En algunos casos, especialmente en algoritmos que requieren ajuste de hiperparámetros, se puede reservar un tercer conjunto denominado validación (Validation set). Sin embargo, en este trabajo no se ha utilizado un conjunto de validación explícito, ya que el objetivo es evaluar el rendimiento del modelo de forma realista a partir de datos temporales.

Evaluación de Modelos de Clasificación

En problemas de clasificación (ej., predecir si un equipo ganará o perderá), se utilizan métricas basadas en la matriz de confusión. [25] La matriz de confusión presenta cuatro valores fundamentales:

- Verdaderos Positivos (TP): Casos correctamente clasificados como positivos.
- Falsos Positivos (FP): Casos incorrectamente clasificados como positivos.
- Verdaderos Negativos (TN): Casos correctamente clasificados como negativos.
- Falsos Negativos (FN): Casos incorrectamente clasificados como negativos.

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Imagen 4– Matriz de Confusión

Métricas de clasificación

Accuracy: Mide el porcentaje de predicciones correctas

- Problema: No es confiable para conjuntos de datos desbalanceados

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: Indica cuántos de los casos clasificados como positivos son realmente positivos.

- Importante cuando los falsos positivos son costosos (ej., detección de fraudes)

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

Recall (sensibilidad o tasa de verdaderos positivos): Indica cuántos de los casos positivos reales fueron correctamente identificados.

- Importante cuando los falsos negativos son críticos (ej., detección de enfermedades).

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

F1-score: Es el promedio armónico entre *Precisión* y *Recall*, ofreciendo una métrica balanceada.

- Ideal para conjuntos de datos desbalanceados

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Balanceo de clases

En problemas de clasificación, es común que las clases no estén representadas de manera equitativa en el conjunto de datos, lo que se conoce como desbalanceo de clases. [12] Cuando una clase es mucho más frecuente que las demás, los modelos de Machine Learning pueden inclinarse hacia la predicción de la clase mayoritaria, afectando la capacidad de generalización y reduciendo la efectividad de las métricas de evaluación.

La **accuracy** (precisión global) es una métrica que se utiliza comúnmente para evaluar modelos de clasificación. Esta métrica mide el porcentaje de predicciones correctas. Sin embargo, en conjuntos de datos desbalanceados, esta métrica suele no ser la más adecuada.

Por ejemplo, si un equipo tiene un 90% de victorias y solo un 10% de empates o derrotas, un modelo que siempre prediga la clase mayoritaria obtendrá un 90% de accuracy, aunque nunca

prediga correctamente la clase minoritaria. Para detectar y evaluar correctamente el impacto del desbalance, es recomendable fijarse en otras métricas como precision y recall.

- Precision y Recall: Permiten evaluar cuántos de los casos predichos como positivos son realmente positivos y cuántos casos reales positivos fueron correctamente identificados.

Estrategias para el balanceo de clases

Para corregir el desbalance, se pueden aplicar diferentes estrategias:

- Undersampling: Reduce la cantidad de instancias de la clase mayoritaria para equilibrar el conjunto de datos. Su método principal incluye la eliminación de instancias de la clase mayoritaria de manera aleatoria, o basado en heurísticas que identifican los ejemplos menos representativos o redundantes.
- Oversampling: El Oversampling aumenta la cantidad de muestras de la clase minoritaria para equilibrar el conjunto de datos. Algunas técnicas comunes incluyen el duplicado de datos de la clase minoritaria de manera aleatoria o la generación de instancias sintéticas interpolando con ejemplos cercanos.

Validación Cruzada

La validación cruzada es una técnica que se utiliza para evaluar el rendimiento de un modelo y garantizar su capacidad de generalización a datos no vistos. [11] En lugar de depender de una única partición de datos, se divide el conjunto en múltiples subconjuntos y se realizan varias iteraciones de entrenamiento y prueba. La variante más utilizada es la k-fold cross validation, en la que los datos se dividen en k partes y el modelo se entrena y valida k veces, alternando el subconjunto destinado a validación. Finalmente, se calcula el promedio de las métricas obtenidas.

En este trabajo, la validación cruzada se ha utilizado exclusivamente durante la fase de optimización de hiperparámetros, concretamente en modelos como KNN, Random Forest y Gradient Boosting, a través del método GridSearchCV. Esta técnica realiza validaciones internas únicamente dentro del conjunto de entrenamiento, sin acceder al conjunto de validación secuencial final.

Dado que los datos del estudio están ordenados cronológicamente (cada fila representa un partido), aplicar validación cruzada sobre el conjunto completo podría provocar fugas de información, al mezclar partidos pasados y futuros en el entrenamiento y validación. Por esta

razón, el enfoque principal empleado para la evaluación final de los modelos es la validación secuencial acumulativa, donde cada predicción se realiza únicamente con los partidos anteriores disponibles, simulando un entorno realista de predicción en tiempo real.

2.3.4 Machine Learning: Modelos Lineales

Los modelos lineales forman la base del aprendizaje automático debido a su simplicidad y eficacia en muchos contextos. Estos modelos asumen que la variable de salida se explica como una combinación lineal de las variables de entrada. En su forma más básica, la ecuación que define este tipo de modelos es:

$$y \approx \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

donde $\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$ son los parámetros (o pesos) que el modelo aprende del conjunto de datos. Para problemas de clasificación, uno de los modelos más utilizados y el que se empleará inicialmente en este trabajo es la **regresión logística**.

Regresión Logística

Un modelo de regresión logística modela la probabilidad de que una instancia pertenezca a una clase determinada. La variable respuesta y es categórica (binaria o multinomial). [13]

Regresión logística para clasificación binaria

La regresión logística se utiliza para tareas de clasificación. Para el caso binario, $y \in \{0,1\}$, y el modelo busca ajustar:

$$\hat{p}(\, y = 1 \mid x\,) = \sigma(w^{\top x} + b)$$

Donde

w es el vector de pesos $w = (\beta_1, \beta_2, \ldots, \beta_n)$

b es el sesgo (bias) o término independiente β_0

$\sigma(\cdot)$ es la función sigmoide dada por $\sigma(z) = \frac{1}{1+i^{-z}}$

Con esta función sigmoide, el valor de la salida se mantiene en el rango (0,1) interpretándose como una probabilidad.

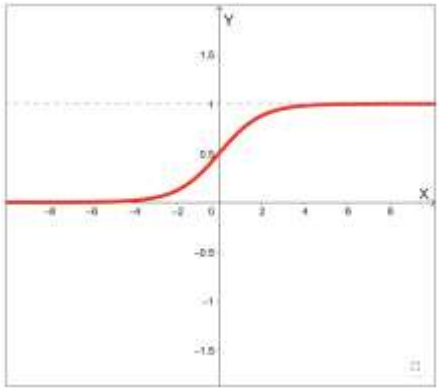

Imagen 5– Representación de la función sigmoide

Regresión logística para clasificación multiclase

La regresión logística es conocida tradicionalmente como un modelo para clasificación binaria, pero existen 2 enfoques para extenderla a clasificación multiclase (por ejemplo, para predecir si un equipo gana, pierde, o empata.

- **One-vs-Rest** (OvR): Para *K* clases se entrenan *K* clasificadores de regresión Logística, uno para cada clase versus las *K-1* restantes. Durante la inferencia, se escoge la clase con mayor probabilidad estimada.
- **Softmax Regression** (o Regresión Logística Multinomial): Se entrenan todas las clases simultáneamente, con una única función de costo *cross-entropy* modificada para la clasificación multinomial. Se generaliza la función sigmoide a la función softmax.

El método Softmax tiende a ser más elegante y consistente matemáticamente, ya que maneja en un único paso las *K* categorías.

Función de costo

La función de costo es una métrica que cuantifica el error entre las predicciones de un modelo y los valores reales. Para entrenar el modelo se utiliza la función de costo de entropía cruzada, también llamada log-loss.

La diferencia entre la función de pérdida (mencionada anteriormente) y la función de costo es que la función de pérdida mide el error para **una sola muestra** del conjunto de datos, mientras la función de costo es el promedio o suma de las pérdidas de **todas las muestras** en el conjunto de datos. En la práctica, ambos términos se confunden porque muchos algoritmos de optimización aplican la misma lógica tanto para una muestra como para el conjunto completo.

Entrenamiento

Para el entrenamiento de la Regresión Logística se hace típicamente mediante gradiente descendiente (o variantes). El objetivo es encontrar w y b que minimicen la función de costo *cross-entropy*

1. Inicialización de los parámetros: Los pesos y el sesgo se inicializan (por ejemplo, en cero o con valores aleatorios).
2. Cálculo de gradientes: Se deriva la función de costo respecto a cada parámetro β_j
3. Actualización de los parámetros según la regla:

$$\beta_j \leftarrow \beta_j - \alpha \cdot \frac{\partial J}{\partial \beta_j}$$

donde α es la tasa de aprendizaje.

Limitaciones

A pesar de su utilidad, los modelos lineales presentan varias limitaciones. Su principal restricción es que asumen una relación lineal entre las variables de entrada y la salida, lo cual no siempre refleja la complejidad de los datos reales. En contextos como la predicción de resultados deportivos, donde influyen numerosos factores interdependientes y no lineales, esta suposición puede reducir considerablemente la capacidad predictiva del modelo. Además, los modelos lineales tienen dificultades para capturar interacciones entre variables o efectos no evidentes sin realizar una ingeniería de características explícita. Por estas razones, resulta necesario comparar su rendimiento con modelos más flexibles que puedan adaptarse mejor a la naturaleza compleja e impredecible del fútbol.

2.3.5 Machine Learning: Modelos no Lineales

Los modelos no lineales de Machine Learning permiten capturar relaciones complejas en los datos que no pueden ser representadas mediante una combinación lineal de las variables independientes. Estos modelos son útiles cuando los datos presentan patrones entre variables que no siguen una tendencia lineal simple. A continuación se describen algunos de los modelos no lineales más utilizados en tareas de clasificación multiclase y regresión.

K-Nearest Neighbors (KNN)

KNN es un algoritmo de aprendizaje supervisado que se puede utilizar tanto para clasificación como para regresión. [14] Es un método de aprendizaje perezoso *(lazy learning)*, lo que significa que no construye específicamente un modelo durante la fase de entrenamiento, sino que guarda el conjunto de datos completo y realiza los cálculos en el momento de la predicción.

Funcionamiento

El principio básico es que la clase o el valor de una muestra se predice en función de sus k vecinos más cercanos en el espacio de características. La distancia entre dos puntos $x_i = (x_i1, x_i2, \dots x_in)$ y $x_j = (x_j1, x_j2, \dots, x_jn)$se calcula utilizando la distancia euclidiana:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2}$$

También se pueden utilizar otras métricas como la distancia Manhattan o la distancia de Minkowski.

- Para clasificación: La clase de una muestra se determina por votación mayoritaria entre los k vecinos más cercanos.
- Para regresión: La predicción se realiza calculando el promedio de los valores numéricos de los k vecinos más cercanos.

Ventajas y desventajas

K-NN es sencillo de implementar y eficaz a la hora de identificar relaciones complejas. Sin embargo, su rendimiento puede disminuir con grandes conjuntos de datos debido a la necesidad de calcular distancias para todas las muestras. También es sensible a la escala de las variables y al ruido en los datos. La elección del número de vecinos k es crucial para evitar el sobreajuste o la subestimación de patrones importantes.

Árboles de Decisión

Los Árboles de Decisión también se pueden utilizar tanto para clasificación como para regresión. [15] La estructura de un árbol de decisión consiste en nodos internos que representan condiciones o preguntas sobre los atributos de entrada, ramas que representan el resultado de estas condiciones, y hojas que representan las predicciones finales (clase o valor).

Funcionamiento

El algoritmo construye el árbol dividiendo recursivamente el conjunto de datos en subconjuntos más pequeños basados en características. Estas características proporcionan la mayor ganancia de información o la mayor reducción de impureza. Hay diferentes medidas de impureza. A continuación se presentan las más comunes:

1. Índice Gini para clasificación:

$$Gini = 1 - \sum_{i=1}^{C} p_i^2$$

   donde p_i es la proporción de instancias de la clase i en un nodo y C es el número de clases

2. Entropía:

$$Entropy = -\sum_{i=1}^{C} p_i \log_2(p_i)$$

3. Error cuadrático medio (MSE) para regresión

El criterio de división busca minimizar estas métricas de impureza, eligiendo la característica y el umbral que mejor separen los datos.

Ventajas y desventajas

Los árboles de decisión son fáciles de interpretar y visualizar. Pueden manejar tanto datos numéricos como categóricos sin necesidad de preprocesamiento. Sin embargo, tienen una alta tendencia al sobreajuste si no se controla la profundidad del árbol o el número mínimo de muestras por hoja. Además, pueden ser inestables frente a pequeñas variaciones en los datos.

Random Forest

El Random Forest es un modelo de *ensemble learning* basado en la combinación de múltiples árboles de decisión para mejorar la precisión y reducir el sobreajuste. [16] Fue propuesto en 2001 y se ha convertido en uno de los algoritmos más robustos y utilizados en la práctica.

Funcionamiento

El Random Forest crea una "colección" o "bosque" de árboles de decisión, cada uno entrenado en una muestra aleatoria del conjunto de datos original (con reemplazo).

1. Muestreo aleatorio con reemplazo: Para cada árbol, se selecciona una muestra aleatoria del conjunto de entrenamiento.
2. Selección aleatoria de características: En cada nodo del árbol, se selecciona un subconjunto aleatorio de características para decidir la mejor división, lo que introduce diversidad entre los árboles.
3. Agregación de resultados:
    a. Clasificación: Se realiza mediante votación mayoritaria entre todos los árboles.
    b. Regresión: La predicción final es el promedio de las salidas de todos los árboles.

Ventajas y desventajas: Random Forest ofrece alta precisión y es robusto frente al sobreajuste, gracias a la diversidad introducida en la construcción de los árboles. Además, es resistente al ruido y puede manejar grandes cantidades de datos y características. Sin embargo, su principal desventaja es la menor interpretabilidad en comparación con un único árbol de decisión y su mayor demanda de recursos computacionales y memoria.

Gradient Boosting

El Gradient Boosting es otra técnica de *ensemble learning* que mejora el rendimiento de los modelos mediante la combinación secuencial de múltiples árboles de decisión. [17] A diferencia de Random Forest, que construye árboles de manera independiente, el Gradient Boosting entrena los árboles secuencialmente, donde cada nuevo árbol intenta corregir los errores cometidos por el árbol anterior.

Funcionamiento

El algoritmo minimiza una función de costo mediante el enfoque de descenso por gradiente.

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

donde

$F_m(x)$ es el modelo después de m iteraciones

$h_m(x)$ es el árbol de decisión entrenado para corregir los errores del modelo anterior

γ_m es la tasa de aprendizaje que controla cuánto contribuye cada nuevo árbol al modelo final.

Entrenamiento

1. Inicialización: Se empieza con un modelo base simple.
2. Cálculo de residuos: Se calcula el residuo o el gradiente negativo del error para cada observación.
3. Entrenamiento del árbol: Se entrena un nuevo árbol para predecir estos residuos.
4. Actualización del modelo: Se actualiza el modelo sumando la predicción ajustada por la tasa de aprendizaje.
5. Repetición: El proceso se repite hasta alcanzar un número predeterminado de iteraciones o hasta que la mejora en la función de costo sea mínima.

Variantes de Gradient Boosting

Existen varias implementaciones del algoritmo clásico de Gradient Boosting que Optimizan el rendimiento y la eficiencia computacional:

- **XGBoost** (Extreme Gradient Boosting): Versión optimizada de Gradient Boosting que incluye mejoras en la velocidad y el rendimiento. Utiliza técnicas de regularización L1 y L2 para reducir el sobreajuste, procesamiento en paralelo y manejo eficiente de los datos faltantes. [18] Implemente un algoritmo llamado **pruning** para eliminar ramas innecesarias en los árboles y mejorar la generalización.

Ventajas y desventajas

Gradient Boosting ofrece un buen rendimiento y permite un ajuste preciso a través de múltiples hiperparámetros. Sin embargo, su naturaleza secuencial puede llevar a tiempos más largos de entrenamiento y un riesgo mayor de sobreajuste si no se regula adecuadamente la complejidad del modelo. También requiere una selección de hiperparámetros adecuada para obtener el mejor rendimiento.

2.3.6 Machine Learning: Redes Neuronales

Las Redes Neuronales son un tipo de algoritmo inspirado en la estructura y funcionamiento del cerebro humano. Están compuestas por unidades llamadas neuronas artificiales, organizadas en

capas y conectadas entre sí mediante pesos ajustables. En este estudio se van a abordar dos tipos de Redes Neuronales, las Redes Neuronales Artificiales y las Redes Neuronales Recurrentes. Las Redes Neuronales Artificiales (ANN) sirven para la identificación de patrones complejos en los datos, mientras que las Redes Neuronales Recurrentes se utilizan para analizar datos secuenciales o temporales.

La Neurona Artificial

Una Neurona Artificial es el componente básico de una red neuronal. Su funcionamiento se basa en un modelo matemático que simula como las neuronas biológicas reciben, procesa y transmiten información. El funcionamiento de una neurona puede representarse de la siguiente forma:

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

donde

x_i representa las entradas o características del dato

w_i son los pesos sinápticos que modulan la importancia de cada entrada

b es un sesgo que permite ajustar el umbral de activación

z la combinación lineal de las entradas y los pesos

A continuación, el valor z pasa por una función de activación, donde a es la salida de la neurona.

Funciones de activación

Transformación matemática aplicada a la salida de una neurona en una red neuronal para introducir no linealidad al modelo. Esto permite a la red aprender y representar relaciones complejas en los datos. Las funciones de activación más comunes son las siguientes:

- Función Sigmoide Convierte la salida en un valor entre 0 y 1, útil para problemas de clasificación binaria.

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

- Función ReLU (Rectified Linear Unit): Introduce no linealidad y es eficiente computacionalmente, evitando problemas de desvanecimiento del gradiente.

$$\mathrm{ReLU}(z) = \max(0, z)$$

- Función Tangente Hiperbólica (tanh): Escala la salida entre -1 y 1, centrándola en cero, para mejorar la convergencia.

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

Arquitectura de una ANN (Artificial Neural Network)

Las ANN están estructuradas en varias capas:

1. Capa de Entrada: Recibe los datos originales
2. Capas Ocultas: Transforman las entradas mediante combinaciones lineales y funciones de activación. El número de capas y neuronas determina la capacidad del modelo para aprender patrones complejos.
3. Capa de Salida: Proporciona la predicción final. En un problema de clasificación binaria, puede tener una sola neurona con activación sigmoide.

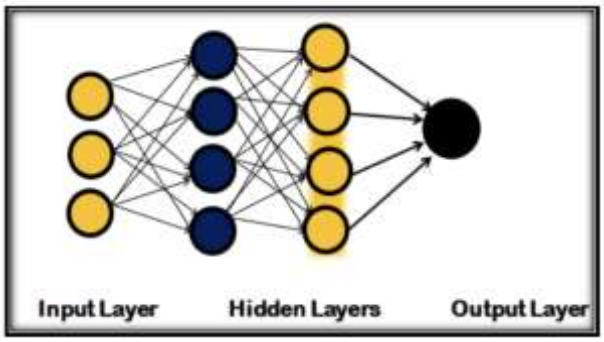

Imagen 6– Red Neuronal Artificial

Entrenamiento de una ANN – Backpropagation

El proceso de entrenamiento de una ANN implica ajustar los pesos y sesgos para minimizar una función de pérdida. [19] Esto se realiza mediante un algoritmo conocido como retropropagación (backpropagation) combinado con descenso de gradiente.

1. Cálculo de la pérdida: Se mide la diferencia entre la predicción "$\hat{y}$" el valor real "y" usando una función de pérdida como el Error Cuadrático Medio (**MSE**)
2. Cálculo del gradiente: Se computan las derivadas parciales de la función de pérdida con respecto a cada peso y sesgo.
3. Actualización de los pesos: Se ajustan los pesos en la dirección que minimiza la pérdida:

$$w_i \leftarrow w_i - \alpha \frac{\partial \mathcal{L}}{\partial w_i}$$

donde α es la tasa de aprendizaje

2.4 Trabajos Relacionados

El uso de modelos de aprendizaje automático en la predicción de resultados deportivos ha sido un área de creciente interés en la última década, con aplicaciones que van desde el análisis de rendimiento deportivo hasta la optimización de estrategias en apuestas. A continuación se presentan algunos estudios relevantes que hablan sobre problemáticas similares.

Herbinet (2018) [20], en su trabajo "*Using Machine Learning Techniques to Predict the Outcome of Professional Football Matches"*, explora diferentes métodos de Machine Learning para predecir el marcador de los partidos de fútbol. En su estudio se introduce una métrica llamada *expected goals* (*xG*), que estima el rendimiento de los equipos mediante eventos ocurridos durante el partido. Algunos de los modelos utilizados en este trabajo son:

1. Modelos Generalizados Lineales: para predecir resultados binarios (victoria/derrota).
2. Árboles de Decisión y Random Forest: utilizados para identificar patrones complejos en el rendimiento de los equipos.
3. Redes Neuronales: aplicadas para capturar patrones no lineales en los datos.

Los resultados mostraron que estos modelos alcanzaron precisiones comparables a las utilizadas por las casas de apuestas, mejorando la capacidad predictiva en comparación con métodos estadísticos tradicionales.

Por otro lado, el estudio *"Aplicación de Métodos de Aprendizaje Automático en el Análisis y la Predicción de Resultados Deportivos"* de Soto-Valero (2018) [21] ofrece una revisión exhaustiva del uso de técnicas de *Machine Learning* en el análisis cuantitativo de datos deportivos. Se destaca el uso de modelos como:

1. Regresión Logística y Árboles de Decisión para la clasificación de resultados competitivos.
1. Redes Neuronales para la evaluación del rendimiento deportivo.
2. Algoritmos de Agrupamiento como K-Means y DBSCAN para la identificación de patrones de comportamiento en el rendimiento de los equipos.

Además, el autor propone una metodología para aplicar estos métodos en el análisis de mercados deportivos, incluyendo las apuestas, lo que resulta directamente relevante para este trabajo.

En un contexto más reciente, Martínez Arias y Marulanda Vélez (2023) realizan una monografía titulada *"Modelo de Clasificación Multiclases para la Predicción de Apuestas Deportivas"* [22]. En este estudio los autores se enfocan en la predicción de resultados en la Serie A de Italia. Utilizan diferentes algoritmos de clasificación, evaluando su precisión mediante diversas métricas:

1. Decision Tree Classifier: obtuvo una precisión del 65%, sirviendo como modelo base.
2. Random Forest Classifier: mejoró la precisión al 73%, reduciendo el sobreajuste mediante la combinación de múltiples árboles.
3. Gradient Boosting Classifier y LightGBM: alcanzaron una precisión del 75%, mejorando iterativamente los errores de modelos anteriores.
4. Histogram Gradient Boosting Classifier: logró una precisión del 75%, destacándose por su eficiencia en el procesamiento de grandes volúmenes de datos.

Baboota y Kaur (2019) [26] realizaron un análisis exhaustivo de la Premier League inglesa, utilizando datos de 11 temporadas (2005–2016) para predecir los resultados de los partidos con distintos modelos de Machine Learning. Compararon algoritmos como regresión logística, Random Forest, K-NN, SVM y XGBoost, siendo este último el que alcanzó el mejor rendimiento, con una precisión del 58.5%. El estudio destaca la importancia de una buena selección de características y de la calibración del modelo, sobre todo en contextos como las apuestas deportivas.

Asimismo, Alfredo et al. (2020) [27] aplicaron modelos de árbol de decisión en la Premier League utilizando una combinación de estadísticas de partidos y técnicas de reducción de características. Compararon los modelos C5.0, Random Forest y XGBoost, siendo el Random Forest el más efectivo, con una precisión del 68.5%, seguido muy de cerca por XGBoost. Estos resultados refuerzan la tendencia observada en la literatura: los modelos de ensemble learning, en particular Random Forest y Gradient Boosting, ofrecen una mejora sustancial frente a técnicas más simples, como los árboles individuales.

En general, la literatura muestra que las técnicas de Machine Learning pueden alcanzar entre 55% y 75% de acierto en predicciones de partidos de fútbol, dependiendo del enfoque, los datos utilizados y si se considera el empate como categoría separada. Modelos más complejos o ensambles tienden a mejorar ligeramente la precisión, aunque el margen de mejora es limitado por la aleatoriedad del deporte.

Estos estudios sirven como base fundamental para el desarrollo del presente TFG, que busca evaluar la capacidad de diferentes modelos de Machine Learning en la predicción de resultados deportivos y su posible aplicación en el ámbito de las apuestas.

3. ASPECTOS METODOLÓGICOS

Este capítulo justifica las técnicas empleadas para investigar, escribir y desarrollar la parte práctica del trabajo. Se detallarán las metodologías utilizadas, las razones para su elección y las tecnologías aplicadas. Este apartado también describe las herramientas y tecnologías utilizadas en el desarrollo del trabajo, justificando su selección en base a eficiencia, compatibilidad y experiencia previa.

3.1 Metodología

El desarrollo del Trabajo de Fin de Grado (TFG) se organizó en diferentes fases, cada una con objetivos definidos. Esta estructura permitió dividir el trabajo en bloques manejables y facilitar su seguimiento y documentación.

3.1.1 Planificación

Inicialmente, se consideraron varios temas para el TFG. Tras un análisis conjunto con la tutora, se seleccionó este tema por su viabilidad, disponibilidad de datos históricos y posibilidad de aplicar las técnicas de Machine Learning aprendidas durante la carrera. Desde el inicio, se establecieron las siguientes fases:

1. Planificación: Se definieron los objetivos del proyecto y se estableció un esquema de trabajo basado en fases. Se llevó a cabo una primera reunión con la tutora para evaluar la viabilidad del tema y se determinó la metodología a seguir.

2. Investigación: Se realizó un estudio de trabajos previos y modelos utilizados en la predicción de resultados deportivos mediante Machine Learning. Se identificaron las principales técnicas aplicadas y se seleccionaron aquellas más relevantes para el trabajo.

3. Adquisición, análisis y procesamiento de datos: Se recopilaron los datos desde Football-Data.co.uk.[8] Durante esta fase, se realizó una limpieza exhaustiva de los datos para corregir inconsistencias, eliminar valores nulos y estructurar un dataset homogéneo.

[8] Football-Data.co.uk. (2024). Spain football data files. https://www.football-data.co.uk/spainm.php

4. Desarrollo del modelo: Se implementaron y compararon distintos modelos de Machine Learning, iniciando con modelos básicos y avanzando hacia modelos más complejos como Random Forest y técnicas de Boosting. Se utilizaron herramientas como MLflow y Dagshub para registrar experimentos y métricas.

5. Evaluación y análisis de resultados: Se analizaron las métricas de rendimiento de los modelos seleccionados y se realizaron comparaciones para determinar la solución más efectiva. Se identificaron patrones en los errores y se evaluaron las limitaciones del modelo aplicado.

A medida que se avanzaba en cada fase, se redactaba la memoria correspondiente a esa etapa, asegurando la coherencia y calidad del documento final. Además, se realizaban revisiones continuas de las secciones previas para garantizar la consistencia. Las fases 3 y 4 se explicarán en detalle en el apartado 4. Desarrollo del Trabajo, y la fase 5 en el apartado 5. Conclusiones.

3.1.2 Organización y seguimiento del trabajo

El trabajo se organizó a través de objetivos semanales, permitiendo una monitorización continua del progreso y una reevaluación constante de los siguientes pasos. Esta metodología ayudó a mantener un ritmo de trabajo estable y evitar desviaciones del objetivo principal. Para asegurar un seguimiento efectivo, se emplearon las siguientes tecnologías:

- GitHub para el almacenamiento y control de versiones del código
- Dagshub y MLflow para registrar los experimentos y las métricas de rendimiento de los modelos

Las reuniones con la tutora se realizaron aproximadamente cada mes, aumentando la frecuencia a medida que se acercaba la fecha de entrega. Antes de cada reunión, se enviaban los avances para su revisión, lo que permitió recibir feedback estructurado y discutir los próximos pasos de manera efectiva

Estas herramientas facilitaron la documentación del proceso y aseguraron la reproducibilidad del trabajo. Las tecnologías empleadas se explicarán con detalle en el siguiente apartado 3.2 Tecnologías Empleadas.

3.1.3 Adaptaciones y ajustes en la metodología

Inicialmente, se consideró implementar tanto la predicción de clases (victoria, empate, derrota) como la predicción de resultados exactos. Sin embargo, esto requería explorar un número

elevado de modelos y técnicas, por lo que se decidió enfocar todos los esfuerzos en un solo objetivo para optimizar su rendimiento. Además, la predicción de resultados exactos es altamente complicada, ya que requiere información histórica sobre el rendimiento detallado de los equipos y jugadores, la cual no está disponible para todas las temporadas. Adicionalmente, no se disponía de las cuotas exactas para cada resultado específico, lo que impedía realizar una validación precisa en relación con las casas de apuestas.

3.2 Tecnologías Empleadas

Este apartado describe las herramientas y tecnologías utilizadas en el desarrollo del trabajo.

3.2.1 Lenguajes y entornos de desarrollo

- Python: Lenguaje principal para la implementación de modelos de Machine Learning y procesamiento de datos.
- Jupyter Notebook: Entorno interactivo utilizado para el desarrollo, análisis exploratorio y pruebas de modelos.

3.2.2 Bibliotecas y Frameworks

- Pandas y NumPy: Manipulación y procesamiento de datos.
- Matplotlib: Visualización de datos y análisis exploratorio.
- Scikit-learn: Implementación de modelos de Machine Learning como Regresión Logística, Random Forest y Gradient Boosting.
- Imbalanced-learn (SMOTE, RandomUnderSampler): Técnicas de balanceo de datos.

3.2.3 Github

GitHub es una plataforma utilizada para el control de versiones y la gestión del código fuente. Permite llevar un registro detallado de los cambios en el código, facilitando el seguimiento de la evolución del proyecto y la posibilidad de volver a versiones anteriores en caso de errores o modificaciones no deseadas. Su uso ayuda a mantener la organización del trabajo, evitar la pérdida de información y colaborar de manera eficiente en proyectos de cualquier escala. Además, al ser accesible desde cualquier dispositivo, permite la gestión remota del código y la integración con otras herramientas utilizadas en el desarrollo de software y ciencia de datos.

3.2.4 MLflow

MLflow es una herramienta utilizada para la gestión y seguimiento de experimentos en Machine Learning. Permite registrar ejecuciones de diferentes modelos, comparar sus métricas de rendimiento y organizar los experimentos de manera eficiente. En este proyecto, se utilizó junto con Dagshub para documentar los experimentos y facilitar la comparación entre modelos con distintas configuraciones.

Dagshub es una plataforma especializada en la gestión de versiones y colaboración en ciencia de datos. Funciona de manera similar a GitHub, pero optimizada para proyectos de Machine Learning, permitiendo almacenar y versionar datasets, modelos y experimentos. Una de sus ventajas es que integra MLflow de manera nativa, lo que facilita el almacenamiento y acceso remoto a los experimentos.

En este caso, el repositorio de Dagshub está directamente vinculado al repositorio del proyecto en GitHub, que actúa como el sistema principal de almacenamiento del código. Esta integración permite que cualquier actualización en GitHub se refleje automáticamente en Dagshub, asegurando que los datos, modelos y experimentos estén correctamente sincronizados y versionados.

La imagen a continuación muestra la interfaz de MLflow en Dagshub, donde se pueden ver múltiples experimentos registrados. Cada fila representa una ejecución diferente de un modelo con sus respectivas métricas y parámetros. Se asignaron etiquetas a cada experimento para monitorizar el rendimiento por equipos. Permite un registro estructurado de experimentos, evitando pruebas desorganizadas.

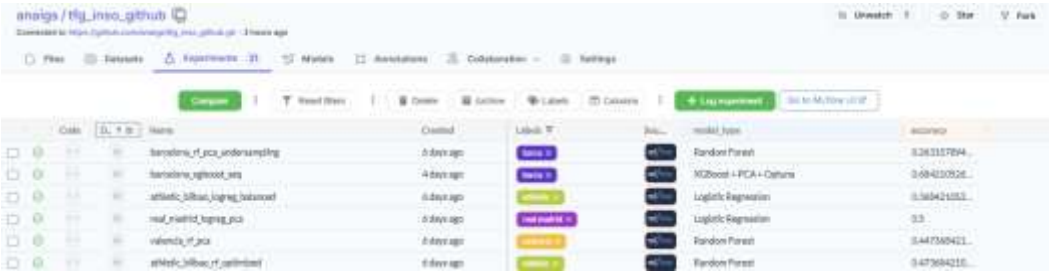

Imagen 7 – Interfaz de MLflow en Dagshub

MLflow permite realizar comparaciones entre los experimentos seleccionados. Se generan gráficos visuales de las métricas que permiten la comparación del rendimiento de los modelos

de manera. La siguiente imagen muestra un diagrama de relaciones entre los modelos y sus métricas.

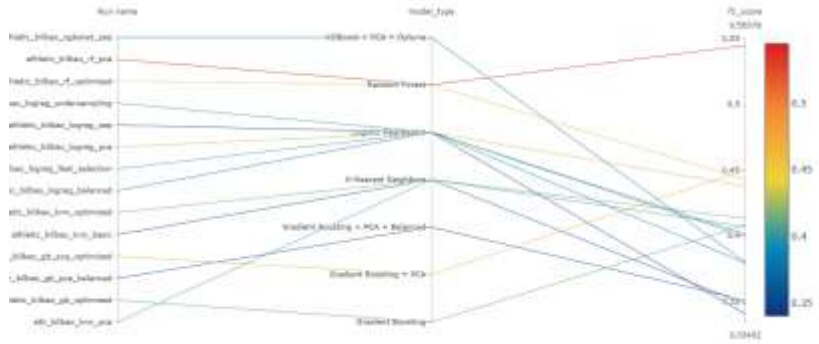

Imagen 8 – Grafico de rendimiento que produce MLflow al comparar modelos

4. DESARROLLO DEL TRABAJO

En este capítulo se detallan las diferentes fases del desarrollo del TFG, describiendo el proceso seguido para alcanzar los objetivos planteados. En particular, en este apartado se abordarán las siguientes fases:

1. **Adquisición, análisis y procesamiento de datos**: Se explicará cómo se obtuvieron los datos necesarios para el desarrollo del modelo, las técnicas empleadas para su limpieza y estructuración, y los criterios utilizados para garantizar la calidad del dataset.
2. **Desarrollo del modelo:** Se explicarán las diferentes técnicas de Machine Learning utilizadas para la predicción de resultados, así como los resultados obtenidos del rendimiento de cada modelo.

Todo el código desarrollado para este proyecto se encuentra alojado en el repositorio de GitHub: TFG_INSO_GitHub. Además, los experimentos realizados de cada modelo y sus resultados se pueden consultar en la plataforma de DagsHub: TFG INSO DagsHub.

4.1 Adquisición, Análisis y Procesamiento de Datos

En este apartado se describen las distintas etapas seguidas para la adquisición, limpieza y procesamiento de los datos en este trabajo.

Para este proyecto, se recopilaron datos históricos de la liga española LaLiga desde la temporada 2003-04 hasta la última temporada completa, 2023-24. Estos datos fueron obtenidos de la plataforma Football-Data.co.uk, una fuente confiable que proporciona información sobre los resultados y estadísticas de múltiples ligas de fútbol en un formato estructurado CSV.

El objetivo de esta fase fue transformar los datos brutos en un conjunto de datos limpio y estructurado, adecuado para el desarrollo de los modelos predictivos. Para ello, se llevaron a cabo las siguientes etapas:

1. Descripción y preproceso de los datos
2. Análisis exploratorio de los datos
3. Generación de datasets por equipo
4. Análisis de datasets por equipos

En los siguientes apartados se describirá en detalle cada una de estas etapas, comenzando por la descripción y el preproceso de los datos recopilados.

Notebooks utilizados

Para llevar a cabo el procesamiento y análisis de los datos, se emplearon los siguientes notebooks:

1. clean_dataset.ipynb: Limpieza y preprocesamiento de datos.
2. analisis_exploratorio.ipynb: Análisis exploratorio del dataset.
3. generacion_datasets.ipynb: Creación de datasets específicos por equipo.
4. analisis_datasets.ipynb: Evaluación de los datasets generados.

Los cuatro datasets generados corresponden a equipos específicos de LaLiga, seleccionados para evaluar su rendimiento de manera individual.

4.1.1 Descripción y Preproceso de los Datos

La plataforma Football-Data.co.uk proporciona información detallada sobre los partidos disputados en distintas temporadas, incluyendo resultados, estadísticas y cuotas de apuestas. Esta plataforma ofrece datos tanto de primera división como de segunda división, pero en este estudio solo se analizarán los partidos de primera división (LaLiga).

Se ha elegido la temporada 2003-04 como punto de partida porque es la primera que incluye datos históricos sobre las cuotas de apuestas, una variable clave en este estudio. Estas cuotas reflejan la probabilidad implícita que los mercados de apuestas asignan a los posibles resultados de un partido. En este contexto, se utilizan como una fuente externa de información que puede mejorar la precisión de los modelos de Machine Learning en la predicción de resultados de los partidos.

A continuación, se describen las variables incluidas en los archivos descargados de Football-Data.co.uk. Es importante destacar que algunas variables solo están disponibles en ciertas temporadas, por lo que su presencia en el dataset puede variar.

Información básica del partido: Contiene datos esenciales que describen el contexto del encuentro.

- Div: División (SP1 = LaLiga)
- Date: Fecha del partido
- Time: Hora del partido
- HomeTeam: Equipo local
- AwayTeam: Equipo visitante

Resultados del partido: Incluye información sobre el resultado final y parcial del partido.

- FTHG: Goles del equipo local (Full Time Home Goals)
- FTAG: Goles del equipo visitante (Full Time Away Goals)
- FTR: Resultado del partido (H = Victoria local, D = Empate, A = Victoria visitante)
- HTHG: Goles del equipo local al descanso (Half Time Home Goals)
- HTAG: Goles del equipo visitante al descanso (Half Time Away Goals)
- HTR: Resultado al descanso (H, D, A)

Estadísticas avanzadas del partido: Se incluyen diferentes métricas sobre el desempeño de los equipos en el partido. Esta información no está disponible en todas las temporadas.

- HS, AS: Disparos del equipo local y visitante
- HST, AST: Disparos a puerta del equipo local y visitante
- HC, AC: Saques de esquina del equipo local y visitante
- HY, AY: Tarjetas amarillas del equipo local y visitante
- HR, AR: Tarjetas rojas del equipo local y visitante

Cuotas de apuestas: Las cuotas de apuestas indican la valoración del mercado sobre los posibles resultados del partido. No todas estan disponibles para cada partido.

1X2 (ganador del partido):

- B365H, B365D, B365A: Cuotas de Bet365 para local, empate y visitante
- PSH, PSD, PSA: Cuotas de Pinnacle
- WHH, WHD, WHA: Cuotas de William Hill
- AvgH, AvgD, AvgA: Cuotas promedio del mercado

Over/Under (Total de goles en el partido):

- B365>2.5, B365<2.5: Cuotas para más/menos de 2.5 goles en Bet365
- Max>2.5, Max<2.5: Máximas cuotas del mercado para más/menos de 2.5 goles
- Avg>2.5, Avg<2.5: Cuotas promedio del mercado

Hándicap asiático:

- B365AHH, B365AHA: Cuotas de Bet365 para hándicap asiático
- BbMxAHH, BbMxAHA: Máximas cuotas del mercado
- AvgAHH, AvgAHA: Cuotas promedio del mercado

Además, existen otras cuotas de diferentes casas de apuestas como **Sporting Odds, Ladbrokes, Gamebookers, Stanleybet, entre otras**. Sin embargo, las mencionadas anteriormente son las más comunes y están presentes en la mayoría de las temporadas. Para más información sobre las variables de los datasets, se puede consultar el archivo de referencia de la plataforma: Football-Data.co.uk./notes.txt

Estructura y procesamiento del dataset

Los datos fueron recopilados en archivos **CSV**, con un archivo separado por cada temporada. Para su análisis, se realizó un proceso de integración y limpieza de los datos, que incluyó las siguientes etapas:

- Lectura de archivos de cada temporada desde la carpeta correspondiente.
- Unificación de los datos en un único dataset.
- Eliminación de valores irrelevantes
- Corrección de formatos
- Creación de la variable Season, que permite identificar a qué temporada pertenece cada partido.

Una vez consolidado el dataset, se aplicaron diversas técnicas de procesamiento para garantizar su calidad y adecuación para el modelo predictivo:

- Selección de variables clave: No se incluyeron estadísticas del partido como remates, posesión o faltas debido a la inconsistencia en la disponibilidad de estos datos, ya que solo algunas temporadas contaban con esta información. Se optó por priorizar variables más consistentes, como los resultados finales de los partidos y las cuotas de apuestas.
- Se eliminaron las variables que contenían información sobre el resultado al descanso (HTHG, HTAG, HTR), ya que la intención del proyecto es predecir el resultado del partido antes de que comience.
- Promedio de cuotas de apuestas: Dado que en muchos partidos no estaban disponibles todas las casas de apuestas, se calcularon los promedios de las cuotas disponibles para cada partido (AvgH, AvgD, AvgA, AvgAHH, AvgAHA). Esta decisión permitió reducir el número de variables, evitar problemas con datos incompletos y centrarse en una medida más representativa del mercado, sin necesidad de incluir las cuotas individuales de cada operador.
- Uso de datos de BetBrain: Para algunos partidos, no se contaba con información de todas las casas de apuestas, sino solo con los valores proporcionados por BetBrain, una plataforma de comparación de cuotas que calcula valores promedio basados en

múltiples operadores. En estos casos, se usaron los datos de BetBrain para rellenar los valores faltantes.

Finalmente, tras la limpieza y preprocesamiento, el dataset quedó listo para el análisis exploratorio, el cual se abordará en el siguiente apartado. La imagen siguiente muestra la estructura final del dataset tras la limpieza y el preprocesamiento:

Date	Season	HomeTeam	AwayTeam	FTHG	FTAG	FTR	AvgH	AvgD	AvgA	AvgAHH	AvgAHA
2003-08-30	2003-04	Albacete	Osasuna	0	2	A	2.21	3.08	2.99	2.05	1.8
2003-08-30	2003-04	Ath Bilbao	Barcelona	0	1	A	2.64	3.13	2.42	1.77	2.08
2003-08-30	2003-04	Espanol	Sociedad	1	1	D	2.58	3.1	2.48	1.87	1.96
2003-08-30	2003-04	Malaga	Villarreal	0	0	D	2.27	3.08	2.88	2.07	1.78
2003-08-30	2003-04	Real Madrid	Betis	2	1	H	1.38	4.0	7.18	1.94	1.91
...	...	...	...	...	...	...	...	...	...	...	...
2024-05-25	2023-24	Real Madrid	Betis	0	0	D	5.9	2.3	4.03	2.0	1.89
2024-05-26	2023-24	Getafe	Mallorca	1	2	A	3.01	2.51	3.25	2.01	1.88
2024-05-26	2023-24	Celta	Valencia	2	2	D	3.36	2.27	3.59	1.97	1.92
2024-05-26	2023-24	Las Palmas	Alaves	1	1	D	2.75	2.63	3.28	1.81	2.09
2024-05-26	2023-24	Sevilla	Barcelona	1	2	A	2.34	3.94	3.67	2.05	1.83

Imagen 9 – Dataset completo de partidos desde la temporada 03/04 hasta 23/24

4.1.2 Análisis Exploratorio de los Datos

Antes de construir los modelos de predicción, se llevó a cabo un análisis exploratorio de los datos para comprender mejor la estructura del dataset, identificar patrones relevantes y evaluar posibles variables que puedan influir en los resultados de los partidos.

Información general del dataset

El dataset utilizado en este estudio cubre un total de 21 temporadas de LaLiga, desde 2003-04 hasta 2023-24. Durante este período, LaLiga ha mantenido un formato consistente con **20 equipos por temporada**, lo que equivale a **380 partidos** jugados por temporada (19 partidos de ida y 19 de vuelta para cada equipo). En total, el dataset contiene **7.980 partidos** analizados.

Además, en este periodo han competido un total de **41 equipos** en LaLiga, aunque algunos han tenido una presencia más estable que otros. Equipos como Real Madrid, Barcelona, Atlético de Bilbao y Valencia han disputado todas las temporadas incluidas en el análisis, mientras que otros como Albacete, Xerez o Córdoba han tenido participaciones más limitadas debido a ascensos y descensos.

Clasificación final por temporada

Para mejorar el análisis y la capacidad predictiva del modelo, se ha generado un dataset con la clasificación final de cada equipo en cada temporada. Esta información permite identificar

patrones de rendimiento histórico y aporta contexto sobre la fortaleza relativa de cada equipo, lo que puede ser clave en la predicción de resultados.

Distribución de goles por partido

Se observa que la mayoría de los encuentros terminan con entre 1 y 3 goles, siendo los partidos con 0 goles relativamente poco frecuentes. El boxplot permite visualizar la dispersión de los goles y la presencia de valores atípicos. Se observan algunos partidos con marcadores inusuales (más de 8 goles), pero en general, la mayoría de los encuentros se mantiene en un rango de 1 a 4 goles.

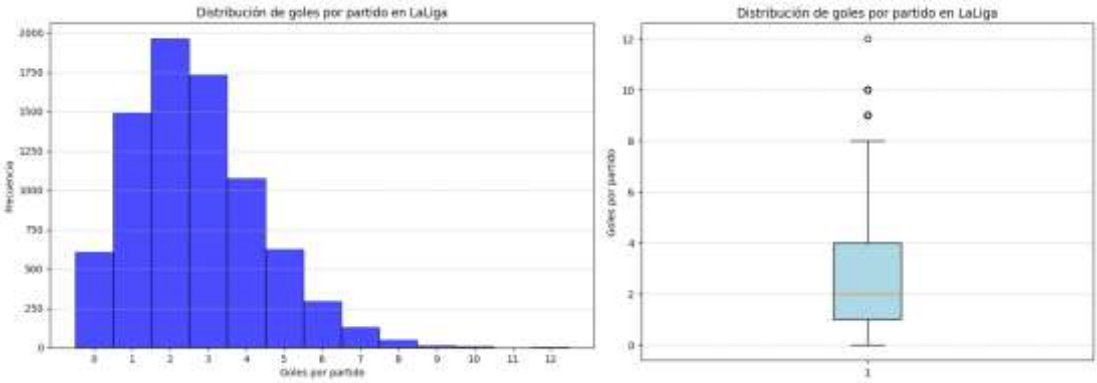

Imagen 10 – Distribución de goles por partido

Rendimiento en casa vs. fuera

El análisis del rendimiento en casa y fuera muestra que la mayoría de los equipos obtienen mejores resultados cuando juegan como locales. No solo ganan más partidos, sino que también marcan más goles y reciben menos en comparación con los encuentros disputados fuera de casa. Esta tendencia es especialmente evidente en equipos de menor regularidad, que dependen más del apoyo local. Aunque equipos como Real Madrid o Barcelona mantienen un rendimiento sólido en ambos escenarios, se observa que para muchos otros el contexto del partido puede marcar una diferencia significativa. Por tanto, la localía se consolida como una variable relevante para tener en cuenta en la predicción de resultados.

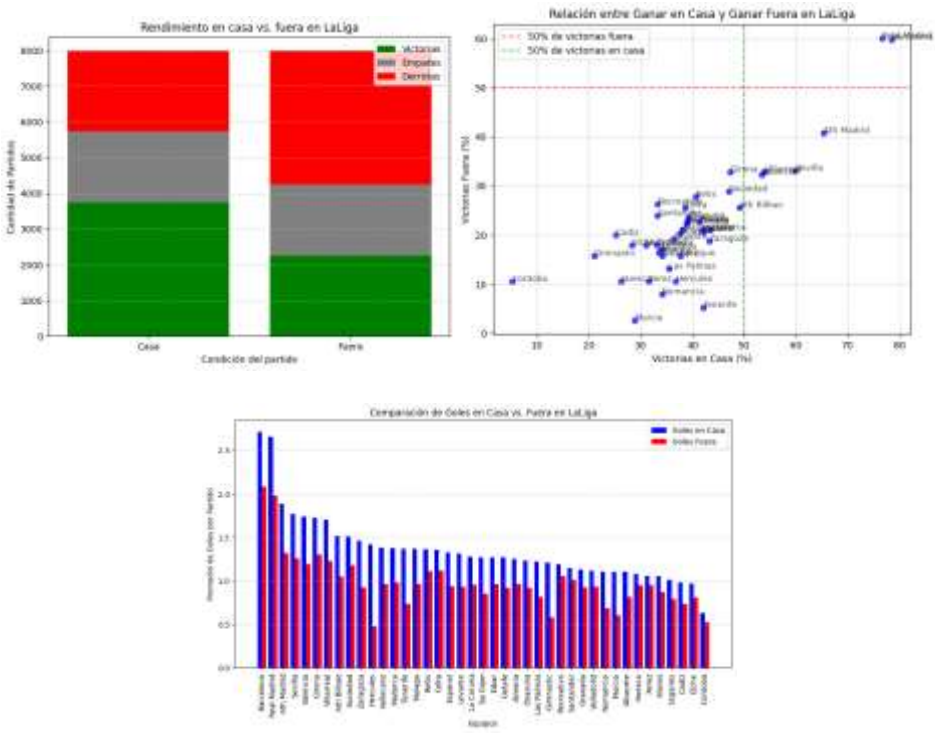

Imagen 11 - Rendimiento en casa vs. fuera

Variabilidad del rendimiento entre temporadas

Este gráfico muestra la cantidad de equipos que han cambiado de posición en la clasificación de una temporada a otra. Se observa que la mayoría de los equipos cambian en promedio 3.32 posiciones por temporada, lo que indica un nivel moderado de estabilidad en LaLiga. Este comportamiento sugiere que la posición final de la temporada anterior podría ser un factor influyente para tener en cuenta en los modelos de predicción.

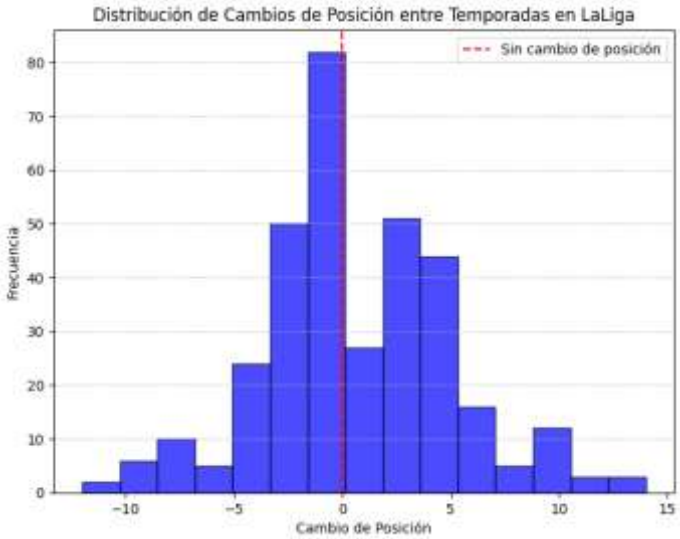

Imagen 12 – Distribución de cambios de posición entre temporadas en LaLiga

4.1.3 Generación de Datasets por Equipos

Después de haber limpiado y transformado los datos, se creó una serie de datasets específicos por equipo. La finalidad de esta etapa es disponer de conjuntos de datos enfocados en equipos concretos, permitiendo un análisis más detallado y la construcción de modelos predictivos adaptados a cada club.

Se seleccionaron cuatro equipos de LaLiga para este estudio:

- Real Madrid
- Barcelona
- Valencia
- Athletic Club de Bilbao

Estos equipos fueron elegidos por su participación constante en LaLiga a lo largo de las temporadas y por el interés de analizar sus tendencias de rendimiento a lo largo del tiempo. Además, el rendimiento de estos equipos en LaLiga varía entre unos y otros. Real Madrid y Barcelona suelen liderar las clasificaciones con un alto porcentaje de victorias, mientras que Valencia y Athletic Club de Bilbao han tenido resultados más variables a lo largo de las temporadas, lo que permite evaluar diferentes tipos de desempeño en el modelo.

Estructura del dataset generado por equipo

Cada dataset creado contiene información detallada sobre los partidos disputados por el equipo en cuestión, considerando tanto su rendimiento general como el de su rival. A continuación, se describen las variables incluidas en estos datasets

Información general del partido: Contexto general del partido y de los equipos:

- season: Temporada en la que se jugó el partido.
- date: Fecha en la que se jugó el partido.
- team: Nombre del equipo analizado.
- rival_team: Nombre del equipo rival.
- home_adv: Indica si el equipo analizado jugó como local (1) o visitante (0).
- last_season_team: Posición final del equipo analizado en la temporada anterior.
- last_season_rival: Posición final del equipo rival en la temporada anterior.

Rendimiento reciente del equipo analizado: Rendimiento del equipo analizado en los últimos 10 partidos de la temporada:

- pct_wins: Porcentaje de victorias en los últimos 10 partidos de la misma temporada.
- avg_goals_scored: Promedio de goles anotados en los últimos 10 partidos.
- avg_goals_received: Promedio de goles recibidos en los últimos 10 partidos.
- goal_difference: Diferencia de goles en los últimos 10 partidos.

Rendimiento reciente del equipo rival: Rendimiento del equipo rival en sus últimos 10 partidos de la temporada:

- pct_wins_rival: Porcentaje de victorias del equipo rival en los últimos 10 partidos.
- avg_goals_scored_rival: Promedio de goles anotados por el equipo rival en los últimos 10 partidos.
- avg_goals_received_rival: Promedio de goles recibidos por el equipo rival en los últimos 10 partidos.
- goal_difference_rival: Diferencia de goles del equipo rival en los últimos 10 partidos.

Historial entre ambos equipos: Rendimiento del equipo analizado frente al rival en los últimos 5 enfrentamientos directos:

- pct_wins_vs_rival: Porcentaje de victorias del equipo analizado contra ese rival en los últimos 5 enfrentamientos.
- avg_goals_scored_vs_rival: Total de goles anotados al rival en los últimos 5 enfrentamientos.
- avg_goals_received_vs_rival: Total de goles recibidos del rival en los últimos 5 enfrentamientos.
- goal_difference_vs_rival: Diferencia de goles en los últimos 5 enfrentamientos.

Cuotas de apuestas: Probabilidad implícita de los resultados según el mercado de apuestas:

- AvgWin: Promedio de cuotas para la victoria del equipo analizado.
- AvgLoss: Promedio de cuotas para la derrota del equipo analizado.
- AvgDraw: Promedio de cuotas para el empate.
- AvgAHWin: Promedio de cuotas para la victoria del equipo analizado con hándicap asiático.
- AvgAHLoss: Promedio de cuotas para la derrota del equipo analizado con hándicap asiático.

Información del resultado del partido: Resultado del partido desde la perspectiva del equipo analizado:

- goals_team: Número de goles anotados por el equipo analizado.
- goals_rival: Número de goles anotados por el equipo rival.
- result: Resultado del partido desde la perspectiva del equipo analizado:
    - Victoria: 1
    - Empate: 0
    - Derrota: -1

Proceso de generación de los datasets por quipo

La generación de datasets específicos por equipo permite analizar el rendimiento individual de cada club y facilita la creación de modelos predictivos personalizados. Gracias a la inclusión de estadísticas de rendimiento reciente, enfrentamientos directos y cuotas de apuestas, estos datasets proporcionan información clave que puede mejorar la precisión en la predicción de los resultados de los partidos.

Para construir estos datasets, se siguieron los siguientes pasos:

1. Filtrado de partidos
    a. Se seleccionaron los partidos donde el equipo analizado participó, ya sea como local o visitante.
2. Asignación de información clave
    a. Se definieron las variables principales, incluyendo la temporada, el equipo analizado, el rival y si el equipo jugaba en casa o fuera.
3. Cálculo de estadísticas de rendimiento
    a. Se calcularon las métricas de los últimos 10 partidos para el equipo y su rival, asegurando que solo se tomaran en cuenta partidos anteriores a la fecha del partido analizado.
    b. Si no se habían jugado aún 10 partidos en la temporada actual, se utilizaron los partidos disponibles, evitando datos de temporadas anteriores para minimizar el impacto de cambios en la plantilla y reflejar el rendimiento real del equipo en ese mismo año.
    c. También se calcularon las estadísticas de enfrentamientos directos entre ambos equipos en los últimos 5 partidos disputados.
4. Ajuste de las cuotas de apuestas
    a. Se aseguraron de que las cuotas siempre correspondieran a la perspectiva del equipo analizado, independientemente de si jugaba como local o visitante.

5. Imputación de valores faltantes
    a. Algunas cuotas de apuestas presentaban valores nulos, por lo que se estableció un valor neutral (1.0) para evitar la pérdida de información.

season	date	team	rival_team	home_adv	last_season_team	last_season_rival	pct_wins	avg_goals_scored
2023-24	2024-05-04	Real Madrid	Cadiz	1	2	38	0.8	2.3
2023-24	2024-05-11	Real Madrid	Granada	0	2	21	0.8	2.2
2023-24	2024-05-14	Real Madrid	Alaves	1	2	21	0.8	2.5
2023-24	2024-05-19	Real Madrid	Villarreal	0	2	5	0.8	2.8
2023-24	2024-05-25	Real Madrid	Betis	1	2	6	0.8	3.1

avg_goals_received	goal_difference	pct_wins_rival	avg_goals_scored_rival	avg_goals_received_rival	goal_difference_rival	pct_wins_vs_rival	avg_goals_scored_vs_rival	avg_goals_received_vs_rival
0.7	16.0	0.2	0.8	1.5	-7.0	0.0	8.0	2.0
0.7	15.0	0.2	1.0	1.6	-4.0	1.0	13.0	2.0
0.6	19.0	0.4	1.0	0.8	1.0	0.8	13.0	4.0
0.6	23.0	0.7	2.2	1.2	10.0	0.2	7.0	6.0
0.8	23.0	0.4	1.6	1.7	-1.0	0.4	4.0	2.0

goal_difference_vs_rival	goals_team	goals_rival	result	AvgWin	AvgLoss	AvgDraw	AvgAllWin	AvgAllLoss
6.0	3.0	0.0	1.0	7.09	8.0	2.13	2.02	1.88
11.0	4.0	0.0	1.0	3.96	2.52	3.59	1.86	1.98
9.0	5.0	0.0	1.0	11.32	8.43	2.46	2.08	1.84
1.0	4.0	4.0	0.0	3.61	2.34	3.35	1.84	2.02
2.0	0.0	0.0	0.0	5.9	6.03	2.3	2.0	1.88

Imagen 13 - Ejemplo de los últimos cinco partidos del Real Madrid en el dataset

Importancia de estas métricas

Las métricas seleccionadas en la construcción de estos datasets juegan un papel clave en la predicción de los resultados de los partidos:

- El rendimiento reciente del equipo y del rival ayuda a identificar tendencias y evaluar si un equipo atraviesa una racha positiva o negativa.
- El historial entre ambos equipos proporciona información sobre si hay un patrón en los enfrentamientos previos. Algunos equipos pueden tener un historial favorable contra ciertos rivales, independientemente de su desempeño general en la liga.
- Las cuotas de apuestas ofrecen una estimación de la probabilidad de cada resultado desde la perspectiva del mercado, lo que puede mejorar la precisión del modelo al incorporar información externa basada en análisis de expertos.

4.1.4 Análisis de Datasets por Equipos

Una vez generados los datasets individuales para cada equipo, se procedió a su análisis con el objetivo de identificar patrones y características relevantes que puedan influir en los resultados de los partidos. Este análisis exploratorio también permite evaluar si las variables creadas aportan información útil y diferenciadora que pueda mejorar el rendimiento de los modelos de predicción.

Evolución del rendimiento a lo largo del tiempo

Se puede observar cómo equipos como Barcelona y Real Madrid mantienen una consistencia alta en ambas métricas, con ligeras fluctuaciones. En cambio, Athletic y Valencia presentan curvas más irregulares, con temporadas fuertes seguidas de otras más débiles.

Estos análisis permiten detectar ciclos de rendimiento y cambios en la competitividad de los equipos a lo largo del tiempo, lo cual puede ser clave para ajustar los modelos predictivos al contexto de cada temporada. Además, es posible que los equipos con un rendimiento más estable a lo largo de los años faciliten la identificación de patrones y, por tanto, obtengan mejores resultados en los modelos de predicción.

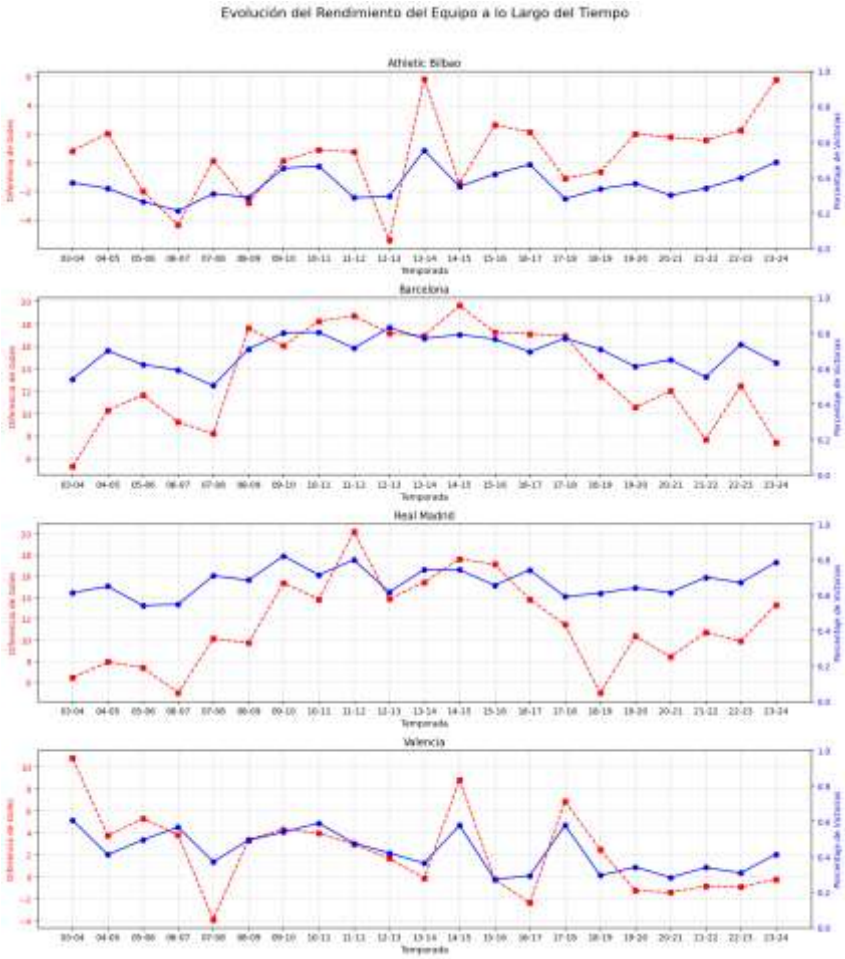

Imagen 14 – Evolución del Rendimiento por Temporadas

Producción y comportamiento ofensivo-defensivo de los equipos

Para evaluar el equilibrio entre ataque y defensa de cada equipo, se analizó la distribución de la diferencia de goles por partido. Esta métrica representa la resta entre los goles anotados y los goles recibidos, y permite resumir el rendimiento global del equipo en un único valor.

El gráfico muestra que Real Madrid y Barcelona mantienen una diferencia de goles consistentemente alta y positiva, lo que refleja su dominio tanto ofensivo como defensivo a lo largo de las temporadas. En cambio, Valencia y especialmente el Athletic Club presentan una mayor dispersión, con valores que se reparten de forma más amplia alrededor del cero. Esto indica que tienen temporadas más variables, con alternancia entre buenos y malos resultados.

Esta diferencia de goles es una métrica especialmente útil para los modelos predictivos, ya que proporciona una visión general de la competitividad del equipo en el conjunto de sus partidos, más allá del número de victorias o derrotas.

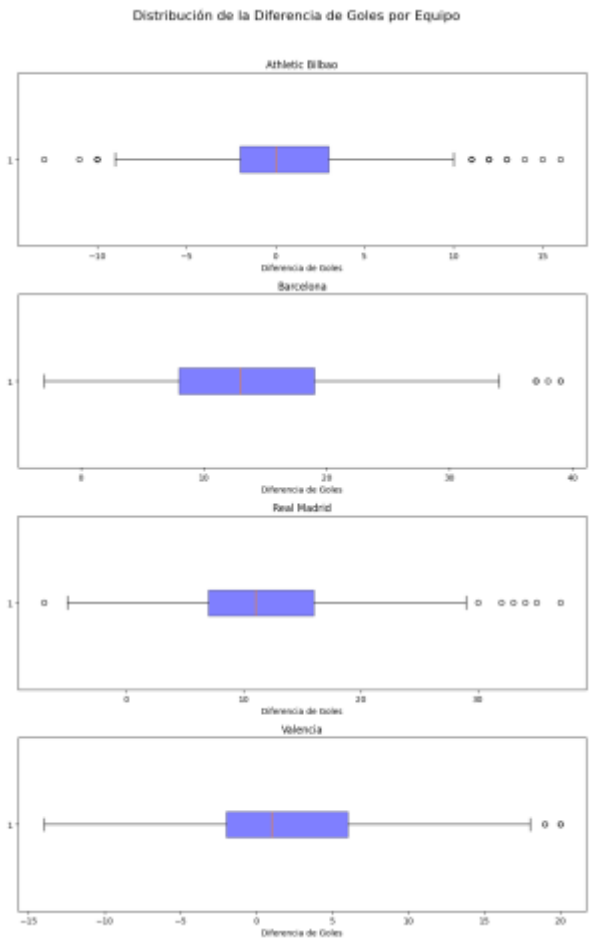

Imagen 15 – Distribución de la Diferencia de Goles por Equipo

Comparativa entre rendimiento en casa y fuera

El gráfico muestra cómo el porcentaje de victorias varía según si los equipos juegan en casa o como visitantes. En los cuatro casos analizados, se observa una ventaja clara al jugar como local. Esta diferencia es más marcada en equipos como el Athletic Club y el Valencia, donde el rendimiento fuera de casa cae notablemente. En cambio, equipos como Barcelona y Real Madrid mantienen una buena proporción de victorias incluso como visitantes, aunque siguen

mostrando mejores resultados en su estadio. Estos datos refuerzan la importancia de tener en cuenta la variable de localía en el desarrollo de modelos predictivos.

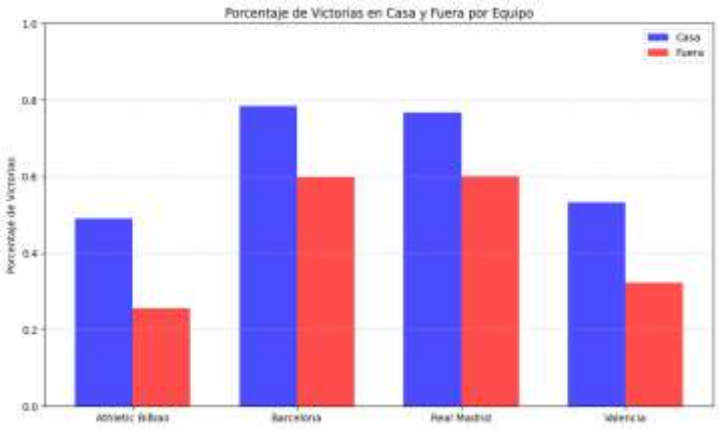

Imagen 16 – Porcentaje de Victorias en Casa y Fuera por Equipo

Relación entre cuotas de apuestas y resultados

Se analizó la relación entre las cuotas de apuestas y los resultados reales de los partidos, encontrando una asociación consistente que refuerza su utilidad como variable predictiva. En general, los equipos tienden a ganar con mayor frecuencia cuando las cuotas asignadas a su victoria son bajas. Esto se refleja en los scatter plots, donde las victorias se agrupan en las zonas con menores cuotas, y también en los boxplots, que muestran una clara diferencia entre las medias de cuotas en partidos ganados y perdidos. Estos patrones sugieren que el mercado de apuestas, en muchos casos, refleja adecuadamente la probabilidad de victoria.

Para cuantificar esta relación, se calculó la correlación entre la cuota de victoria (AvgWin) y el resultado del partido (result). Los coeficientes obtenidos para cada equipo son los siguientes:

- 0.35 en Athletic
- -0.31 en Valencia
- -0.21 en Barcelona
- -0.18 en Real Madrid

En todos los casos, la correlación es negativa, lo que indica que, a medida que aumenta la cuota, disminuye la probabilidad real de que el equipo gane. Aunque esta relación es más débil en equipos como Barcelona y Real Madrid, que suelen ganar incluso con cuotas no tan bajas, el patrón general valida las cuotas como un buen estimador del rendimiento esperado.

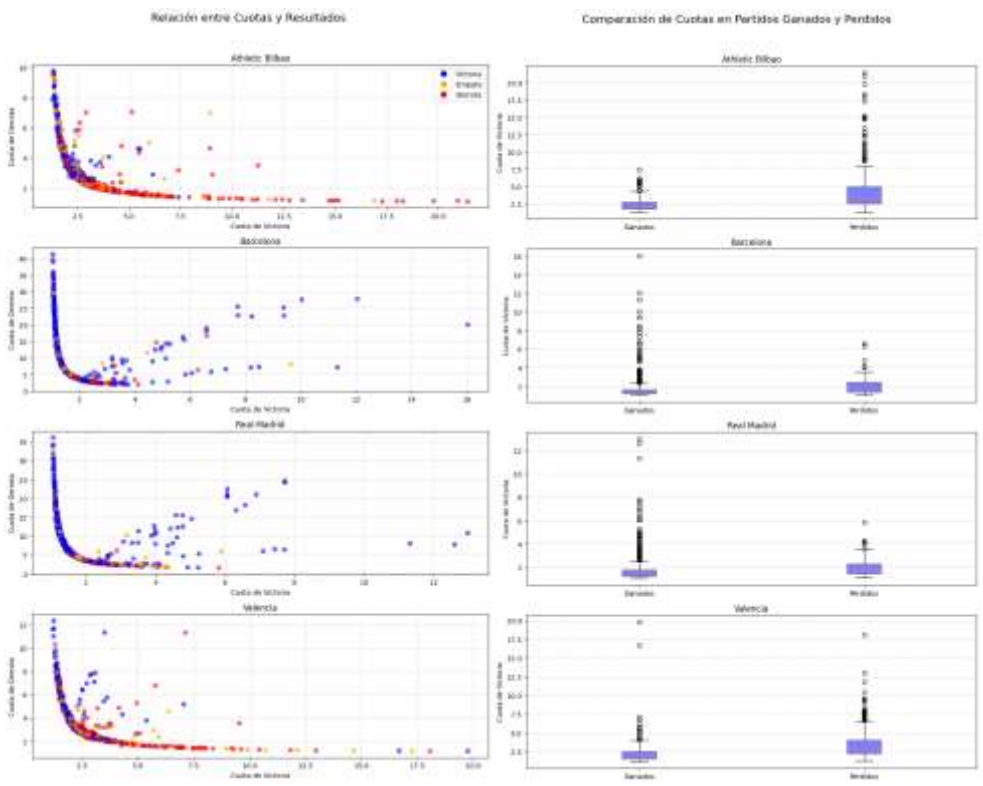

Imagen 17 – Relación entre Cuotas y Resultados

4.2 Desarrollo del modelo

4.2.1 Introducción a los Modelos

En esta sección se detallan los distintos modelos que se han desarrollado para predecir el resultado de partidos de fútbol, utilizando como variable objetivo *result*, que toma los valores 1 (victoria), 0 (empate) y -1 (derrota), siempre desde la perspectiva del equipo analizado.

El objetivo principal del modelo es simular una situación realista de predicción, donde el sistema reciba los datos disponibles antes del partido y pueda anticipar su resultado. Para ello, se ha diseñado un flujo de validación que respeta la naturaleza temporal de los datos y evita fugas de información.

Procesamiento de los datos

Antes de entrenar los modelos, se aplicó un preprocesamiento que incluyó:

- Eliminación de las columnas *goals_team*, *goals_rival* y *result*, siendo esta última la variable objetivo a predecir.

- Codificación de la variable *rival_team* mediante *LabelEncoder*, con el fin de que los modelos pudieran manejarla como variable categórica.
- Escalado de los datos mediante técnicas de normalización, con el objetivo de mejorar la estabilidad numérica y el rendimiento de los modelos.

Conexión a MLflow

Para gestionar y comparar los experimentos realizados durante el desarrollo de los modelos, se incorporó el uso de la plataforma **Mlflow**. En este proyecto, MLflow se ha conectado a través de **DagsHub**, que actúa como servidor remoto para el seguimiento de experimentos. En el siguiente bloque de código se muestra cómo se establece esta conexión:

```python
USERNAME = "anaigs"

REPO_NAME = "tfg_inso_github"

# Configurar el tracking URI con autenticación

mlflow.set_tracking_uri(f"https://dagshub.com/{USERNAME}/{REPO_NAME}.mlflow")

dagshub.init(repo_owner=USERNAME, repo_name=REPO_NAME, mlflow=True)
```

Para ver un ejemplo completo de cómo se registra un experimento en MLflow (incluyendo el registro de métricas, parámetros y artefactos), consultar el Anexo A.

División del conjunto de datos

Para evaluar el desempeño de los modelos se utilizaron dos enfoques complementarios:

1. Validación por temporadas completas: se reservan 38 o 76 partidos (una o dos temporadas) como conjunto de validación, entrenando con el resto de los partidos anteriores. Esta estrategia busca un equilibrio entre tener suficientes datos de entrenamiento y evaluar en condiciones realistas.
2. Validación secuencial acumulativa: este enfoque simula el uso del modelo en un entorno real, prediciendo partido por partido a medida que avanza la temporada. El algoritmo sigue los siguientes pasos:
    1. Se utiliza la última temporada completa como conjunto de validación.
    2. Para cada partido, el modelo se entrena con todos los partidos previos a su fecha.
    3. Se realiza la predicción del resultado.

4. Se añade ese partido al conjunto de entrenamiento, acumulando experiencia.
5. El proceso se repite para el siguiente partido.

Este enfoque permite capturar la dinámica temporal del fútbol, ya que los datos del partido anterior se incorporan antes de hacer la siguiente predicción, reproduciendo el funcionamiento de un sistema en producción. Ver Anexo B para más detalles sobre la implementación.

A continuación se analizan en detalle los cinco modelos utilizados en este trabajo:

1. Regresión Logística
2. K-Nearest Neighbors (KNN)
3. Random Forest
4. Gradient Boosting
5. Redes Neuronales

4.2.2 Regresión Logística

La regresión logística es un modelo de clasificación lineal que ha sido ampliamente utilizado como modelo base en problemas de predicción multiclase. En total, se llevaron a cabo **18 experimentos distintos** de regresión logística, variando tanto los equipos como las técnicas de preprocesamiento y validación aplicadas.

- Validación secuencial (partido a partido)
- Validación por temporadas (38 y 76 partidos).
- Reducción de dimensionalidad mediante PCA.
- Selección de características relevantes.
- Técnicas de balanceo de clases: undersampling y SMOTE (oversampling).
- Combinaciones de preprocesamientos (por ejemplo, undersampling + PCA).

Se utilizó el solver *lbfgs*, un optimizador basado en gradiente adecuado para problemas de clasificación multiclase y de rápida convergencia. Se estableció un número máximo de iteraciones de 500 y se aplicó escalado previo de características.

Justificación

La regresión logística fue seleccionada como modelo base por su facilidad de implementación, rapidez de entrenamiento y utilidad como referencia para modelos más complejos. Sin embargo, los primeros resultados evidenciaron problemas derivados del **desbalance de clases**, especialmente en equipos como Real Madrid y Barcelona, donde predominan las victorias.

Esto provocó situaciones donde el modelo predecía exclusivamente la clase mayoritaria (victoria), alcanzando una accuracy elevada pero un rendimiento real deficiente. En otras palabras, el modelo simplemente aprendía a predecir la clase mayoritaria.

Para abordar este problema se aplicaron técnicas de **balanceo de clases** y se incorporó el uso del **F1-score**, una métrica más adecuada para evaluar el rendimiento equilibrado entre todas las clases.

Resultados

Se obtuvieron resultados variados dependiendo del equipo, la técnica de validación y los ajustes aplicados. La siguiente figura muestra cómo la combinación de PCA con balanceo de clases fue clave para mejorar el rendimiento global del modelo:

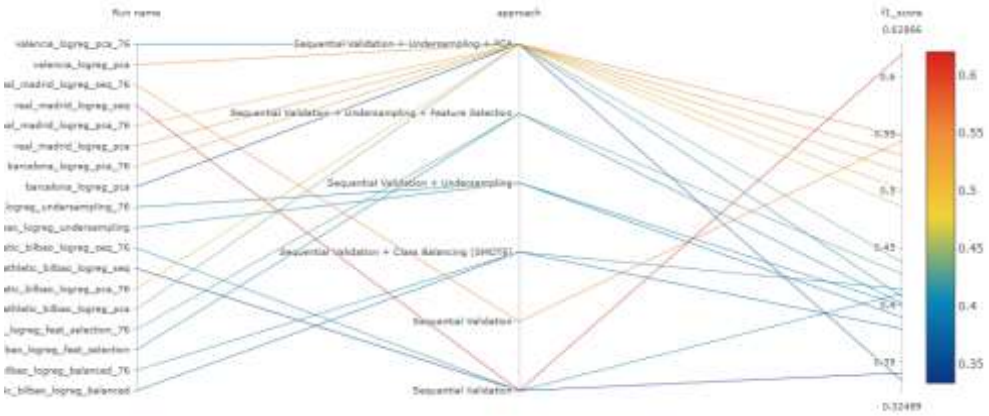

Imagen 18 - Comparativa de *f1_score* entre Experimentos de Regresión Logística

El modelo más exitoso según accuracy fue el de Real Madrid con PCA y validación secuencial, alcanzando un 68%. Sin embargo, esto se explica por el alto porcentaje de victorias en ese conjunto de datos.

```
                                                                                                                    output

Accuracy del modelo en validación secuencial: 0.6842

               precision    recall f1-score   support

Derrota (-1)       0.00      0.00      0.00         1

Empate (0)         0.00      0.00      0.00         8

Victoria (1)       0.74      0.90      0.81        29
```

Como puede observarse, el modelo predice únicamente victorias, ignorando empates y derrotas, lo que invalida su utilidad en un contexto real.

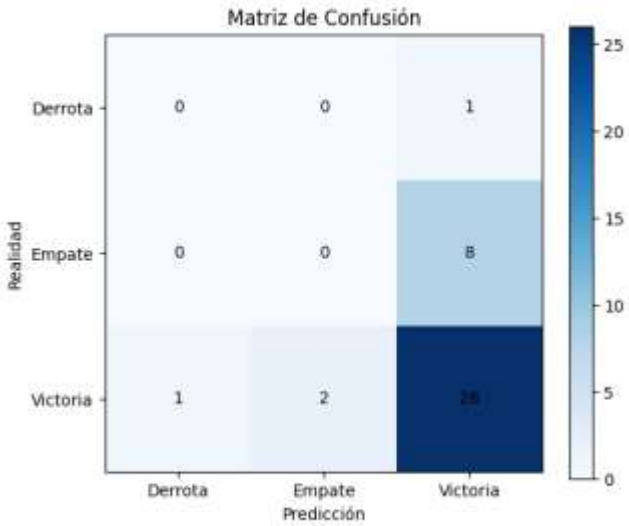

Imagen 19 - Matriz de Confusión del Modelo Básico para el Real Madrid

Posteriormente, se aplicó **PCA** y ***Undersampling*** al mismo conjunto del Real Madrid. Aunque la accuracy bajó al 50%, el modelo mejoró considerablemente su capacidad para predecir otras clases. Aunque se reduce la capacidad del modelo para acertar victorias, mejora notablemente su capacidad para detectar empates y derrotas, ofreciendo un comportamiento más equilibrado.

```
output

Accuracy del modelo con PCA y Undersampling para real_madrid: 0.5000

               precision    recall   f1-score   support

Derrota (-1)       0.00      0.00      0.00         1

Empate (0)         0.30      0.88      0.45         8

Victoria (1)       0.92      0.41      0.57        29
```

Para contrastar, también se utilizó el Athletic Club de Bilbao como equipo de prueba, ya que presenta un historial más equilibrado entre victorias y derrotas, permitiendo una evaluación más realista de las capacidades del modelo.

La combinación de PCA + undersampling + validación secuencial, aplicada en este caso al Athletic Club de Bilbao, logró un F1-score de hasta 0.49, el más alto entre todas las configuraciones probadas para este equipo. Esta estrategia mejoró especialmente la predicción de derrotas y empates, anteriormente mal clasificadas.

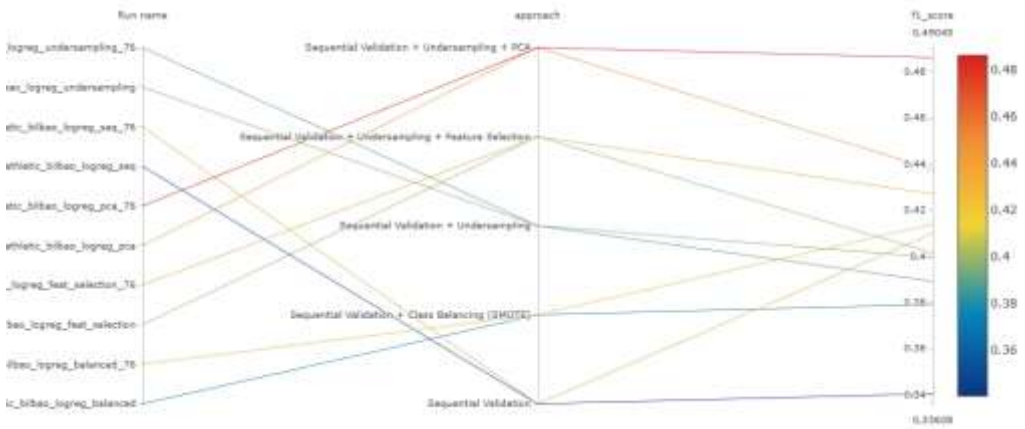

Imagen 20 - Comparativa de Configuraciones y F1-score para Athletic Club de Bilbao

Conclusiones

La regresión logística ha servido como modelo base útil para este proyecto, gracias a su simplicidad y eficiencia. Sin embargo, en equipos como Real Madrid o Barcelona, donde predominan las victorias, la accuracy resulta engañosa al no reflejar correctamente el bajo rendimiento del modelo en clases minoritarias como empates o derrotas.

Por ello, fue necesario aplicar técnicas de balanceo de clases y reducción de dimensionalidad para mejorar el comportamiento del modelo. En este contexto, el F1-score se consolidó como la métrica más adecuada, al ofrecer una evaluación equilibrada entre todas las clases.

El uso del Athletic Club, con un reparto más uniforme de resultados, permitió comprobar que el modelo puede mejorar significativamente cuando se entrena con datos más balanceados. Aunque la regresión logística tiene limitaciones, especialmente ante relaciones no lineales, sigue siendo una opción válida como punto de partida en tareas de predicción multiclase desbalanceadas.

4.2.3 KNN: K – Nearest Neighbors

K-Nearest Neighbors (KNN) es un modelo de clasificación no paramétrico basado en la proximidad entre muestras. En este trabajo se realizaron **6 experimentos** distintos con KNN, aplicando el enfoque de validación secuencial acumulativa y utilizando el conjunto de datos del Athletic Club de Bilbao como caso principal. Se exploraron distintas configuraciones:

- Balanceo de clases mediante undersampling
- Búsqueda de hiperparámetros (k, métrica de distancia) con GridSearchCV.
- Reducción de dimensionalidad con PCA (reteniendo el 95% de la varianza).

- Comparación de rendimiento entre versión básica, con PCA y con hiperparámetros optimizados.
- Los parámetros evaluados incluyeron valores de k entre 3 y 15, y distintas métricas de distancia: Euclidean, Manhattan y Minkowski.

Justificación

El modelo KNN se eligió por su simplicidad y por ser una técnica de referencia útil para problemas de clasificación. Sin embargo, al tratarse de un algoritmo sensible a la escala y al número de vecinos, se aplicó escalado previo y se ajustaron los hiperparámetros para mejorar el rendimiento.

Dado que los datos del Athletic Club presentan una distribución más equilibrada entre clases, se consideró adecuado para evaluar la capacidad del modelo para distinguir entre victoria, empate y derrota. También se aplicó el modelo a otros equipos (Real Madrid, Barcelona y Valencia) para comparar su comportamiento en diferentes contextos competitivos.

Resultados

El rendimiento de KNN fue, en general, **inferior** al esperado. Incluso con ajuste de hiperparámetros y reducción de dimensionalidad, los resultados no superaron valores moderados de F1-score. El mejor experimento fue el modelo optimizado del Athletic Club, con un F1-score de 0.41. En otros equipos, el rendimiento fue incluso más bajo, especialmente en Valencia, donde el F1-score cayó por debajo de 0.30.

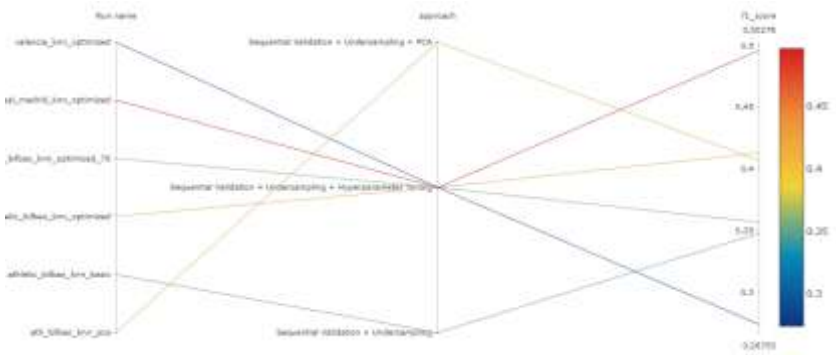

Imagen 21 - Comparativa de F1-score entre distintos enfoques de KNN

Las mejoras más significativas se produjeron al aplicar **PCA o al optimizar los hiperparámetros**, lo que sugiere que el modelo es sensible a la reducción de ruido y a la correcta elección de *k* y la métrica de distancia. Para consultar cómo se ha realizado este proceso de optimización utilizando GridSearchCV, puede consultarse el Anexo C.

En la siguiente imagen se muestra la matriz de confusión correspondiente a la mejor configuración obtenida (KNN optimizado):

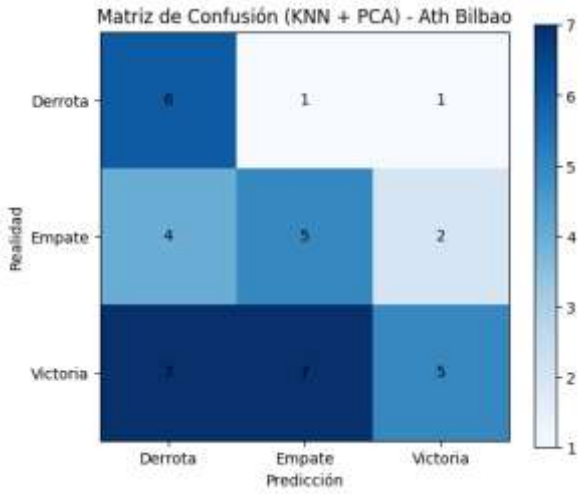

Imagen 22 - Matriz de confusión del modelo KNN optimizado (Athletic Club)

Además, se aplicó el modelo a otros equipos con resultados mixtos:

- Barcelona: Accuracy 0.50, F1-score 0.54
- Real Madrid: Accuracy 0.39, F1-score 0.50
- Valencia: Accuracy 0.29, F1-score 0.27

Las pruebas con conjuntos de validación de 76 partidos no ofrecieron mejoras significativas respecto al tamaño estándar de 38 partidos.

Conclusiones

KNN no ha ofrecido buenos resultados en este problema. Aunque logró cierta mejora al aplicar PCA y ajustar hiperparámetros, su rendimiento general ha sido inferior al de otros modelos probados. Una posible explicación para el bajo desempeño es que KNN es muy sensible a la estructura del espacio de características: si los datos no están claramente separados, o si la información relevante está dispersa, el modelo tiene dificultades para distinguir clases. Además, su comportamiento puede degradarse rápidamente cuando hay desbalance de clases, incluso aplicando técnicas como el undersampling.

El caso de Valencia es especialmente ilustrativo: su rendimiento histórico es muy variable, y su dataset no presenta patrones claros ni estables entre temporadas. Esto dificulta aún más la clasificación por similitud, que es la base del funcionamiento de KNN. Por estos motivos, se descartó continuar explorando variantes adicionales con KNN, priorizando modelos con mejor capacidad de adaptación al problema.

4.2.4 Random Forest

Random Forest es un modelo de tipo ensamblado que combina múltiples árboles de decisión para mejorar la estabilidad y precisión de las predicciones. En este trabajo se probaron distintas configuraciones utilizando validación secuencial acumulativa. Se realizaron un total **de 14 experimentos**, incluyendo pruebas con distintos equipos y técnicas combinadas de preprocesamiento:

- Optimización de hiperparámetros (n_estimators, max_depth, min_samples_split, min_samples_leaf) mediante GridSearchCV. Para ver un ejemplo de cómo se ha realizado este proceso, puede consultarse el Anexo C.
- Reducción de dimensionalidad con PCA, reteniendo el 95% de la varianza.
- Balanceo de clases mediante Random Undersampling.
- Combinaciones de las anteriores: PCA + undersampling, con y sin ajuste de hiperparámetros.
- Validación con test size de 38 y 76 partidos.

Todos los experimentos utilizaron escalado previo de características y validación secuencial, entrenando el modelo partido a partido para simular un escenario realista.

Justificación

Random Forest fue seleccionado por su capacidad para manejar relaciones no lineales, su resistencia al sobreajuste y su buen rendimiento en tareas de clasificación multiclase. A diferencia de modelos más simples como la regresión logística o KNN, Random Forest permite capturar patrones complejos sin requerir una normalización perfecta de las variables.

Resultados

El rendimiento del modelo varió significativamente según el equipo y la configuración. A nivel general, se observaron las siguientes tendencias:

- El modelo más efectivo fue el de Real Madrid con PCA y validación secuencial, alcanzando un accuracy del 71%. No obstante, al igual que en otros modelos, este valor se explica principalmente por la predominancia de victorias en ese equipo.
- La mejor combinación en términos de F1-score equilibrado fue el modelo del Athletic Club, que incluyó PCA y ajuste de hiperparámetros, alcanzando un F1-score de 0.54 con test size 76.
- La combinación de PCA + undersampling también mostró mejoras respecto a la versión básica, especialmente en equipos más desbalanceados como el Real Madrid.

En la siguiente figura se presenta una **comparativa de F1-score** de todos los experimentos realizados con Random Forest:

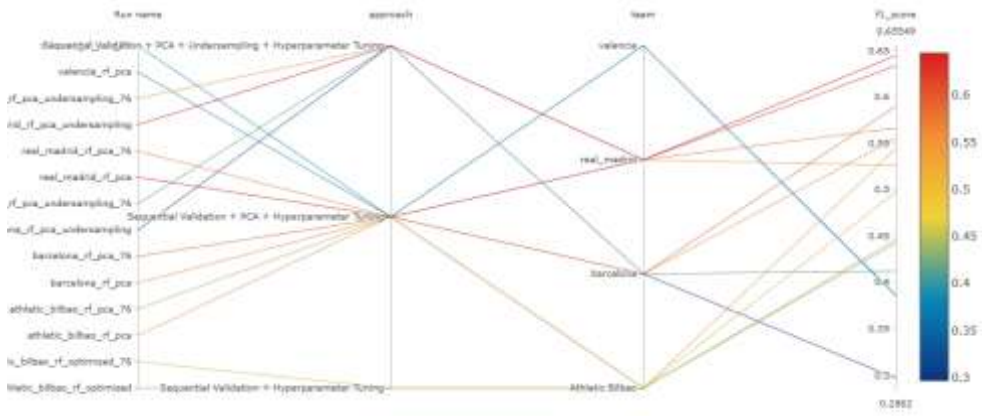

Imagen 23 - Comparativa de F1-score en modelos Random Forest

En general, los modelos con **PCA + optimización ofrecieron mejoras moderadas**, y los experimentos con test size de 76 partidos mostraron mejores resultados solo en algunos casos.

Además, en el caso del Real Madrid, se observó una mejora clara al aplicar técnicas de balanceo: el modelo sin balanceo predecía casi exclusivamente victorias, mientras que al aplicar undersampling se logró una mayor capacidad de detección de empates y derrotas, elevando el F1-score de 0.567 a 0.645.

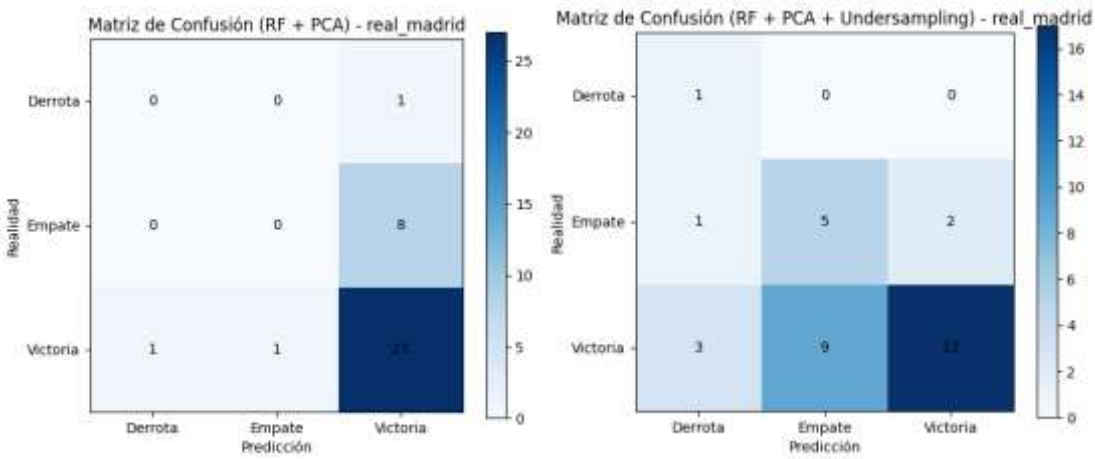

Imagen 24 – Comparativa entre Modelos Random Forest (Real Madrid)

Conclusiones

Random Forest se ha comportado de forma más robusta que modelos como KNN, mostrando mejor capacidad para detectar patrones complejos incluso con datos ruidosos. Su rendimiento fue particularmente sólido en el Athletic Club, donde los resultados están más equilibrados entre clases.

Sin embargo, en equipos como Real Madrid o Barcelona, el modelo tiende a sesgarse hacia la clase mayoritaria si no se aplica balanceo. Las técnicas de PCA y undersampling permitieron contrarrestar parcialmente este problema, pero no aunque no siempre se logró un equilibrio perfecto entre clases.

En conjunto, Random Forest se perfila como una opción potente para este tipo de tareas, especialmente si se acompaña de una adecuada selección de características y estrategias de balanceo.

4.2.4 Gradient Boosting y XGBoost

En esta sección se empezó evaluando el modelo Gradient Boosting, y posteriormente se adoptó XGBoost, una versión más eficiente y regularizada que incorpora técnicas adicionales de penalización y optimización. Se realizaron un total de **21 experimentos**, aplicando diferentes enfoques combinados para mejorar el rendimiento de los modelos.

- Validación secuencial acumulativa (con test size de 38 y 76 partidos).
- Optimización de hiperparámetros con GridSearchCV (para Gradient Boosting, ver Anexo C) y posteriormente con Optuna (para XGBoost, ver Anexo D).
- Reducción de dimensionalidad mediante PCA, conservando el 95% de la varianza.
- Aplicación de técnicas de balanceo de clases, principalmente con SMOTE.
- Combinaciones de los anteriores: PCA + SMOTE, GridSearchCV vs Optuna, y análisis comparativo de rendimiento.

Justificación

Gradient Boosting fue elegido inicialmente por su capacidad para modelar relaciones no lineales y su robustez frente al sobreajuste. Sin embargo, al observar mejoras tanto en velocidad como en precisión con XGBoost, se optó por centrar los esfuerzos en este último. Asimismo, se adoptó Optuna como método de optimización, ya que superó claramente a GridSearchCV en eficiencia y en rendimiento validado. Además, se mantuvieron las técnicas de balanceo de clases y reducción de dimensionalidad mediante PCA, ya que en modelos anteriores (como Random Forest) demostraron contribuir significativamente a la mejora del F1-score.

Resultados

El mejor resultado global se obtuvo con el Real Madrid, utilizando XGBoost con PCA, SMOTE y Optuna, alcanzando un F1-score de 0.683. Le siguieron Barcelona con 0.590, Athletic Club con 0.450 y Valencia con 0.435. En general, los modelos con XGBoost presentaron una ventaja clara respecto a los de Gradient Boosting, tanto en rendimiento global como en estabilidad.

Una de las mejoras más significativas se observó al aplicar técnicas de balanceo con SMOTE, particularmente en equipos con una distribución de clases muy desigual. En el caso de Real Madrid, por ejemplo, el modelo sin SMOTE presentaba un fuerte sesgo hacia las victorias, con una precisión del 69% en esa clase, pero prácticamente nula en derrotas y empates. Esto se reflejó en un F1-score global de 0.560, con un recall del 0.00 para derrotas y del 0.07 para empates.

```
                                                                    output

Real Madrid - Modelo SIN SMOTE:

Accuracy: 0.6316

               precision    recall  f1-score   support

Derrota (0)       0.00      0.00      0.00         9

Empate (1)        0.17      0.07      0.10        14

Victoria (2)      0.69      0.89      0.78        53
```

Tras aplicar SMOTE, el modelo logró una mejora considerable en estas clases minoritarias, alcanzando un F1-score de 0.624. El recall para derrotas y empates aumentó a 0.33 y 0.36 respectivamente, y aunque se sacrificó ligeramente el accuracy (de 0.63 a 0.61), el resultado fue un modelo más equilibrado y representativo del comportamiento real del equipo.

```
                                                                    output

Real Madrid - Modelo CON SMOTE:

Accuracy: 0.6053

               precision    recall  f1-score   support

Derrota (0)       0.21      0.33      0.26         9

Empate (1)        0.33      0.36      0.34        14

Victoria (2)      0.81      0.72      0.76        53
```

En la siguiente figura se presenta una comparativa de F1-score entre todos los modelos de Gradient Boosting y XGBoost evaluados. Se puede observar que los modelos basados en XGBoost con Optuna y SMOTE (líneas rojas en la imagen) dominaron consistentemente en términos de rendimiento general:

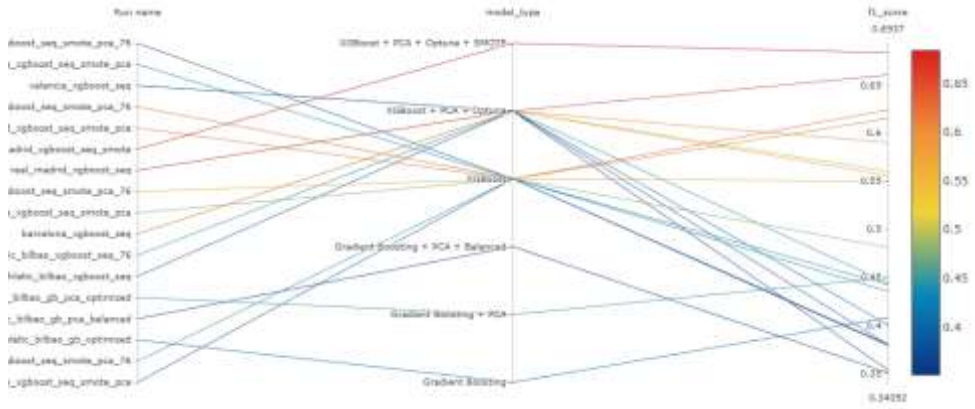

Imagen 25 - Comparativa de F1-score entre modelos Boosting

Con el objetivo de comprobar la efectividad real de SMOTE, se llevó a cabo un seguimiento de la distribución de clases generada en cada iteración de la validación secuencial. Este análisis permitió verificar que SMOTE estaba funcionando adecuadamente, generando conjuntos balanceados y evitando el predominio artificial de alguna clase, especialmente en equipos como Real Madrid o Barcelona, donde el desbalance era más acusado.

Conclusiones

Los modelos basados en XGBoost destacaron por ofrecer un rendimiento superior en comparación con otras técnicas exploradas, como KNN o incluso Gradient Boosting. Gracias a su capacidad para capturar relaciones no lineales complejas, sus mecanismos internos de regularización y su eficiencia en el entrenamiento, XGBoost demostró ser especialmente adecuado para abordar problemas de clasificación multiclase en el contexto deportivo.

Los mejores resultados se alcanzaron mediante la combinación de XGBoost con PCA, SMOTE y Optuna. Esta configuración sugiere que un enfoque integral, que considere tanto la naturaleza del modelo como las particularidades del conjunto de datos, es clave para mejorar el desempeño. La reducción de dimensionalidad mediante PCA ayudó a eliminar ruido y variables redundantes, mientras que el balanceo de clases con SMOTE mejoró notablemente la capacidad del modelo para predecir empates y derrotas, clases menos frecuentes en la mayoría de los equipos. Por su parte, Optuna permitió optimizar los hiperparámetros de forma más eficiente y eficaz que GridSearchCV, lo que resultó especialmente valioso en escenarios con un espacio de búsqueda amplio.

No obstante, el rendimiento varió en función del equipo analizado. Equipos como el Real Madrid, con patrones más estables y una distribución de clases menos problemática, facilitaron el aprendizaje del modelo. En contraste, otros conjuntos como el Valencia, con mayor variabilidad entre temporadas, plantearon una mayor dificultad para obtener predicciones fiables. Esto evidencia la importancia de adaptar los modelos no solo a nivel técnico, sino también considerando las particularidades estadísticas y deportivas de cada conjunto.

4.2.4 Redes Neuronales

En esta sección se exploró el uso de Redes Neuronales como alternativa para modelar resultados deportivos. Dado que este tipo de modelos requieren **mayor capacidad computacional**, se realizaron un total de **6 experimentos** con distintas configuraciones, centrados principalmente en el equipo Athletic Club, además de pruebas con Barcelona, Real Madrid y Valencia. Se probaron diferentes estrategias combinadas:

- Validación secuencial acumulativa (test size = 76 partidos).
- Aplicación de técnicas de regularización como Dropout y EarlyStopping.
- Uso de Optuna para la optimización de arquitectura e hiperparámetros.
- Evaluación del impacto de PCA como reducción de dimensionalidad.
- Ajuste de pesos con class_weight para abordar el desbalance entre clases.

Justificación

El interés en redes neuronales radica en su capacidad para capturar patrones no lineales complejos. No obstante, requieren un volumen de datos y tiempo de entrenamiento considerablemente mayor. Por este motivo, se utilizó **Google Colab,** aprovechando sus recursos de cómputo, especialmente durante el proceso de optimización con Optuna. Para ver un ejemplo de cómo se ha realizado este proceso, puede consultarse el Anexo D.

Resultados

El mejor resultado general se obtuvo con Athletic Club utilizando red neuronal optimizada con PCA, alcanzando un F1-score de 0.52. Le siguieron Real Madrid con 0.46, Valencia con 0.43 y Barcelona con 0.34 en sus mejores configuraciones. En la siguiente imagen se observa la comparación de F1-score entre los distintos enfoques evaluados:

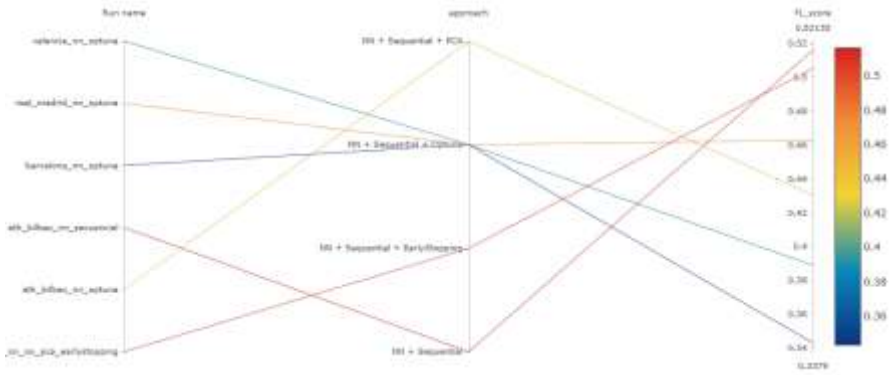

Imagen 26 - Comparativa de F1-score entre modelos con redes neuronales

Los experimentos muestran que el uso de Optuna resultó especialmente útil para ajustar la arquitectura de las redes neuronales, permitiendo encontrar modelos más eficientes en menos tiempo que con métodos tradicionales como GridSearch. En cambio, la reducción de dimensionalidad con PCA no siempre aportó mejoras: en algunos casos incluso perjudicó ligeramente el rendimiento, por lo que no se aplicó de forma generalizada. Por su parte, la estrategia de ajustar los pesos de clase (class_weight) fue clave para mejorar la capacidad del modelo a la hora de predecir clases menos representadas, como los empates, sin necesidad de modificar directamente los datos de entrenamiento.

Conclusiones

Aunque las redes neuronales ofrecieron un rendimiento aceptable en algunos escenarios, los resultados obtenidos fueron generalmente inferiores a los de modelos como XGBoost. Una posible razón es que, dada la cantidad limitada de datos y la alta variabilidad entre equipos y partidos, las redes neuronales no lograron generalizar de forma consistente sin caer en sobreajuste. Además, la sensibilidad del modelo a la distribución de clases y su dependencia de configuraciones finas (como el número de capas, unidades y dropout) dificultaron obtener mejoras sustanciales sin un coste computacional elevado.

4.2.5 Mejores Resultados

La siguiente tabla resume los mejores modelos obtenidos para cada uno de los cuatro equipos analizados, junto con su respectivo F1-score. Como puede observarse, XGBoost fue el modelo que ofreció un rendimiento más alto en dos de los casos (Real Madrid y Barcelona), mientras que para Athletic de Bilbao el mejor resultado se obtuvo con Random Forest, y para Valencia, con Regresión Logística.

	Real Madrid	Barcelona	Athletic de Bilbao	Valencia
Modelo	XGBoost	XGBoost	Random Forest	Regresión Logística
F1-score	0.684	0.590	0.547	0.548

Figura 1 - Mejores modelos por equipo según F1-score, considerando todas las configuraciones evaluadas en el estudio.

Es posible que para el caso de Valencia el modelo de Regresión Logística haya ofrecido el mejor rendimiento por varias razones relacionadas con las características del equipo y la naturaleza de sus datos. Ya durante las primeras etapas de preprocesamiento se observa que el historial del equipo presenta una mayor variabilidad entre partidos, sin una tendencia clara o un patrón dominante de resultados como ocurre en equipos más estables como el Real Madrid. Esta falta de regularidad puede limitar la capacidad de modelos complejos como XGBoost o redes neuronales para aprender patrones significativos, llevándolos a un mayor riesgo de sobreajuste. En cambio, modelos más simples como la Regresión Logística pueden generalizar mejor bajo estas condiciones, priorizando relaciones lineales que resultan más robustas frente al ruido.

Finalmente, en el apartado 5. CONCLUSIONES, se presentará una comparativa más detallada entre los distintos modelos utilizados, evaluando no solo su rendimiento general, sino también su comportamiento frente al desbalance de clases y su capacidad de adaptación a diferentes estructuras de datos.

4.2.6 Conclusión General de los Resultados

A lo largo de este proyecto se han llevado a cabo **65 experimentos** que combinan diversos modelos, técnicas de preprocesamiento y estrategias de validación. Se ha trabajado con cuatro equipos (Real Madrid, Barcelona, Athletic de Bilbao y Valencia) para comprobar la solidez de los algoritmos en escenarios con diferente distribución de victorias, empates y derrotas.

Inicialmente, se recurrió a la métrica *accuracy* para evaluar el rendimiento de los modelos. No obstante, al observar un claro desbalance en la distribución de clases (victoria, empate y derrota), se advirtió que dicha métrica ofrecía una imagen distorsionada del desempeño, ya que tendía a favorecer la clase mayoritaria. Ante esta limitación, se exploraron métricas alternativas

y se optó por el uso del *F1-score*, al ser más apropiado en contextos con clases desiguales. Esta métrica, que combina precisión (*precision*) y exhaustividad (*recall*) en una media armónica, proporciona una evaluación más equilibrada del rendimiento. Aunque también se consideró el uso del área bajo la curva ROC (AUC-ROC), estudios como el de Saito y Rehmsmeier (2015) [28] sostienen que, en situaciones con clases desbalanceadas, resulta más pertinente emplear la curva *precision-recall*, ya que aporta información más ajustada a este tipo de escenarios.

A continuación, se muestran los **cuatro mejores modelos** en cada equipo, ordenados de mayor a menor F1-score, incluyendo la posición en el ranking, el tipo de modelo, el F1-score alcanzado, las técnicas de optimización/balanceo empleadas y el tamaño del conjunto de prueba (partidos reservados en la validación final).

Real Madrid

Posición	Modelo	F1-score	Técnicas	Tamaño del conjunto de prueba
1	XGBoost	0.684	PCA, Optuna y SMOTE	38
2	XGBoost	0.661	PCA, Optuna	38
3	Random Forest	0.645	PCA, GridSearchCV, Undersampling	38
4	Random Forest	0.634	PCA, Optuna	38

Figura 2 – Comparativa de los cuatro mejores modelos para el equipo Real Madrid.

FC Barcelona

Posición	Modelo	F1-score	Técnicas	Tamaño del conjunto de prueba
1	XGBoost	0.590	PCA, Optuna	76
2	Random Forest	0.590	PCA, GridSearchCV	76
3	XGBoost	0.556	PCA, Optuna	38

4	Random Forest	0.556	PCA, GridSearchCV	38

Figura 3 – Comparativa de los cuatro mejores modelos para el equipo FC Barcelona.

Athletic de Bilbao

Posición	Modelo	F1-score	Técnicas	Tamaño del conjunto de prueba
1	Random Forest	0.545	PCA, Optuna	38
2	Red Neuronal	0.516	Modelo básico inicial	76
3	Red Neuronal	0.506	Optuna, EarlyStopping, ClassWeights	76
4	Random Forest	0.499	GridSearchCV	76

Figura 4 – Comparativa de los cuatro mejores modelos para el equipo Athletic de Bilbao.

Valencia CF

Posición	Modelo	F1-score	Técnicas	Tamaño del conjunto de prueba
1	Regresión Logística	0.548	PCA, Undersampling	38
2	XGBoost	0.436	PCA, Optuna, SMOTE	38
3	Regresión Logística	0.407	PCA, Undersampling	76
4	XGBoost	0.401	PCA, Optuna	38

Figura 5 – Comparativa de los cuatro mejores modelos para el equipo Valencia CF.

En general, los empates fueron mucho menos frecuentes que las victorias o las derrotas en la mayoría de los equipos de La Liga, lo que tuvo un impacto directo en el rendimiento de los modelos. Esta diferencia se notó especialmente en equipos como el Real Madrid y el Barcelona, donde las victorias eran claramente predominantes. En estos casos, muchos modelos

aprendieron a predecir sobre todo la clase mayoritaria, dejando en segundo plano los empates y las derrotas. Para intentar corregir este desequilibrio, se aplicaron técnicas como *undersampling* y *SMOTE*, que ayudaron a mejorar el reparto de clases y facilitaron la detección de resultados menos habituales.

En equipos con una distribución más repartida, como el Athletic de Bilbao o el Valencia, los modelos consiguieron un comportamiento más equilibrado, aunque seguir acertando los empates fue especialmente complicado. Esto ya se intuía en el análisis exploratorio de los datos, donde se comprobó que los empates eran, con diferencia, la clase menos representada (Imagen 11).

Por otro lado, es interesante que en algunos casos, modelos más simples como la Regresión Logística ofrecieran mejores resultados que otros más complejos como XGBoost o Redes Neuronales. Un buen ejemplo es el Valencia, un equipo con resultados más irregulares. Esa falta de estabilidad puede hacer que los modelos avanzados, al buscar patrones muy específicos, se ajusten demasiado a los datos de entrenamiento y funcionen peor con nuevos datos. En cambio, los modelos sencillos, al tener una estructura más básica, pueden adaptarse mejor cuando hay más variación o ruido en los datos.

Algo similar ocurrió con el Athletic de Bilbao, donde el mejor modelo de red neuronal fue el más simple: una red básica creada al inicio del proyecto, sin optimización de hiperparámetros ni técnicas de análisis de características o clases. Este modelo tenía solo dos capas densas (de 32 y 16 neuronas), activación ReLU y una capa *dropout* del 0.2 para evitar sobreajuste, además de la capa de salida. A medida que se probaban configuraciones más complejas, el rendimiento del modelo fue bajando, lo que refuerza la idea de que más complejidad no siempre significa mejores resultados.

Por el contrario, XGBoost y Random Forest funcionaron especialmente bien en equipos con patrones más claros, como el Real Madrid o el Barcelona, aunque tienen la desventaja de ser más difíciles de interpretar que modelos como la Regresión Logística.

Resultados Globales

- Equilibrio entre clases: Se alcanzaron mejores resultados al aplicar técnicas de balanceo como SMOTE o undersampling junto con PCA, especialmente en equipos con distribuciones muy desiguales.
- Modelos más eficaces con historial dominante: En equipos como Real Madrid y Barcelona, donde predominan las victorias, XGBoost y Random Forest ofrecieron los mejores F1-score tras aplicar técnicas de balanceo.

- Dificultad en la predicción de empates: Los empates fueron la clase más difícil de modelar debido a su baja frecuencia y a la complejidad de los contextos en los que ocurren. Esto ya se observó en el análisis exploratorio.
- Simplicidad frente a complejidad: En equipos con mayor variabilidad como el Valencia o el Athletic, modelos más simples como la Regresión Logística o una red neuronal básica superaron a configuraciones más complejas, que tendían al sobreajuste.

En conjunto, estos resultados reflejan la complejidad de modelar resultados deportivos y la importancia de adaptar las técnicas al tipo de datos y a la realidad de cada equipo. Los aprendizajes extraídos de este análisis servirán como base para las conclusiones generales del trabajo, así como para orientar futuras líneas de investigación.

5. CONCLUSIONES

Esta sección recoge las principales conclusiones del trabajo realizado, evaluando el cumplimiento de los objetivos planteados, analizando críticamente los resultados obtenidos, y proponiendo futuras líneas de investigación.

5.1 Objetivos Planteados y Grado de Cumplimiento

El objetivo principal de este trabajo era explorar y evaluar la capacidad de los modelos de Machine Learning para predecir resultados en un entorno deportivo, donde predomina la alta aleatoriedad. A lo largo del desarrollo del proyecto se han cumplido, en mayor o menor medida, los objetivos específicos planteados inicialmente. A continuación, se detalla el grado de cumplimiento de cada uno

Mejorar la eficiencia de los modelos

Se han aplicado técnicas de preprocesamiento y reducción de la dimensionalidad, como PCA, que han ayudado a extraer información relevante y reducir el ruido en los datos. También se creó un nuevo dataset con variables derivadas tras analizar el conjunto original, y se dividieron los datos por equipos para adaptar mejor los modelos a las particularidades de cada caso. Tras aplicar estas técnicas, se observaron claras mejoras en el rendimiento de los modelos, especialmente al combinarlas con estrategias de balanceo como SMOTE o undersampling.

Desarrollo y evaluación de modelos

Se han implementado diversos algoritmos de clasificación, incluyendo modelos avanzados como XGBoost, Random Forest y Redes Neuronales, así como otros más sencillos como la Regresión Logística. Además, se han explorado distintas técnicas de optimización de hiperparámetros, como GridSearchCV y Optuna, aplicadas en función del modelo y del experimento.

Definición y aplicación de métricas de evaluación

Para evaluar el rendimiento de los modelos, se realizó una transición desde la métrica de accuracy (utilizada en los primeros experimentos) hacia métricas más adecuadas para escenarios con clases desbalanceadas. En este contexto, se priorizó el uso del F1-score, ya que ofrece una medida más equilibrada de la capacidad del modelo para predecir correctamente las tres clases posibles: victoria, empate o derrota. La elección de esta métrica se basó tanto en la naturaleza del problema como en recomendaciones de investigaciones previas.

Predicción de resultados multiclase

Se ha logrado construir modelos capaces de predecir tres posibles resultados (victoria, empate o derrota). Aunque las predicciones de empates han demostrado ser especialmente complejas, los modelos han logrado, en general, distinguir entre las clases con un rendimiento razonable, especialmente tras aplicar técnicas de balanceo.

Comparar la efectividad de los distintos modelos propuestos

Se ha llevado a cabo una comparativa detallada del rendimiento de los modelos en función del equipo analizado. Esto ha permitido observar cómo ciertas configuraciones y algoritmos funcionan mejor en equipos con patrones de resultados más claros (como el Real Madrid o el Barcelona), mientras que en equipos con mayor variabilidad (como el Valencia o el Athletic de Bilbao), modelos más simples han ofrecido mejor desempeño. Para facilitar esta comparación, se ha utilizado la plataforma MLflow, que ha permitido registrar y visualizar de forma estructurada los distintos experimentos, modelos y técnicas aplicadas, facilitando así el análisis comparativo entre configuraciones.

Examinar limitaciones y efectividad

Este objetivo se abordará en detalle en el siguiente apartado 5.3 Limitaciones, donde se expondrán de forma crítica las principales barreras encontradas, así como el potencial real de estas técnicas para reducir la aleatoriedad inherente a los resultados deportivos.

5.2 Limitaciones

Limitaciones técnicas

Durante el desarrollo del proyecto se han identificado varios aspectos metodológicos que podrían haber afectado al rendimiento de los modelos. En primer lugar, el tamaño de los conjuntos de prueba era limitado, especialmente al dividir los datos por equipos, lo que reduce la capacidad generalizadora de los modelos. El hecho de trabajar con cuatro equipos distintos introdujo un reto importante: la gran variabilidad en sus patrones de resultados hacía difícil encontrar una configuración de modelo que funcionara bien en todos los casos. Mientras algunos equipos mostraban comportamientos más estables, otros presentaban una alta irregularidad, lo que obligaba a adaptar las técnicas a cada contexto. Es posible que, utilizando el conjunto de datos completo sin dividir por equipos, se hubiera logrado una mayor

consistencia en el rendimiento global de los modelos, aunque a costa de perder la personalización por equipo que se buscaba en este trabajo.

Además, aunque se exploraron técnicas de optimización como GridSearchCV y Optuna, no todos los modelos fueron ajustados exhaustivamente debido a las restricciones de tiempo y recursos computacionales. En el caso concreto de las redes neuronales, se entrenaron y optimizaron en Google Colab para aprovechar una mayor capacidad de procesamiento. Aun así, no fue posible trabajar con arquitecturas mucho más complejas, ya que el proceso de optimización de estos modelos es especialmente costoso y requiere largos tiempos de ejecución, lo que limitó las posibilidades de experimentación.

También es importante señalar que, a pesar de aplicar técnicas de balanceo como SMOTE o undersampling, la naturaleza del desbalance entre clases seguía dificultando la predicción de los empates, que fueron consistentemente la clase con peor rendimiento.

Limitaciones contextuales

Más allá del modelo, existen factores externos que escapan al control del sistema y que afectan de forma directa al resultado de un partido de fútbol. Variables como lesiones, decisiones arbitrales, cambios tácticos durante el encuentro, el estado anímico de los jugadores o condiciones meteorológicas no están reflejadas en los datos utilizados, lo que limita inevitablemente la precisión de las predicciones. Además, el fútbol es un deporte con un alto componente aleatorio y emocional, por lo que incluso los mejores modelos pueden verse superados por situaciones impredecibles. Finalmente, el hecho de centrarse en cuatro equipos específicos limita la capacidad de generalización del estudio al resto de la liga o a competiciones diferentes.

5.3 Discusión y Análisis Crítico

Fortalezas y debilidades

Una de las principales fortalezas del trabajo ha sido el enfoque estructurado y progresivo en la construcción, evaluación y comparación de modelos. Se ha trabajado con un conjunto diverso de algoritmos, aplicando técnicas de preprocesamiento, balanceo y optimización que han permitido adaptar los modelos a contextos diferentes, como es el caso de cada equipo. También se ha utilizado MLFlow para documentar y comparar de forma clara los resultados, lo que ha facilitado el análisis crítico del rendimiento.

Entre las debilidades, destaca la dificultad de encontrar una configuración que funcionase bien para todos los equipos. La alta variabilidad entre ellos obligó a adaptar los modelos a contextos muy distintos, lo que limitó la posibilidad de aplicar soluciones generalizables. Además, las restricciones técnicas, como el uso limitado de recursos computacionales, impidieron explorar arquitecturas más avanzadas, sobre todo en el caso de redes neuronales.

Fiabilidad de los modelos y comparación con la bibliografía

Los resultados obtenidos en este trabajo están en línea con los encontrados en otros estudios previos. Por ejemplo, Baboota y Kaur (2019) o Alfredo et al. (2020) muestran que modelos como XGBoost y Random Forest suelen alcanzar entre un 58% y 70% de precisión, lo que coincide con los mejores F1-score obtenidos en este estudio. Además, la literatura destaca que modelos de ensemble suelen superar a técnicas más simples, algo que también se ha observado en este trabajo, especialmente en equipos con una distribución de resultados más estable.

Sin embargo, también se ha evidenciado lo que otros autores señalan: la aleatoriedad del fútbol impone un límite natural a la capacidad predictiva de los modelos, y en muchos casos la mejora entre modelos más simples y complejos es modesta. Como en el estudio de Martínez Arias y Marulanda Vélez (2023), aquí también se ha observado que modelos como la regresión logística pueden competir en rendimiento cuando los datos presentan mucha variabilidad.

Implicaciones prácticas

Este trabajo demuestra que el *Machine Learning* puede aportar valor en la predicción de resultados deportivos, especialmente en el ámbito de las apuestas. Sin embargo, el fútbol es un entorno altamente impredecible, y factores como la propia influencia del mercado de apuestas o eventos inesperados durante el partido dificultan alcanzar una precisión elevada. Por ello, aunque este tipo de modelos tienen gran potencial, todavía no se aplican de forma generalizada, ya que sus resultados no son lo suficientemente fiables para su uso directo en entornos reales.

5.4 Propuestas de Trabajos Futuros

Exploración de nuevos métodos

Aunque los resultados obtenidos han sido coherentes con lo observado en la bibliografía y han confirmado el potencial de modelos como XGBoost o Random Forest, habría sido interesante realizar algunos experimentos utilizando el conjunto de datos completo (sin dividir por equipos)

para analizar cómo afecta esto al rendimiento general de los modelos y facilitar comparaciones más globales.

Por otro lado, sería útil explorar algoritmos que tengan en cuenta la evolución temporal de los equipos a lo largo de una temporada, como las redes neuronales recurrentes (*RNN*) o las variantes *LSTM*, que podrían capturar dinámicas de forma más precisa. Además, como línea de investigación más avanzada, se podrían probar modelos basados en arquitecturas *transformer*, que están demostrando buenos resultados en problemas secuenciales, y podrían adaptarse a contextos deportivos si se dispone de datos estructurados temporalmente.

Profundización del enfoque

Para evaluar el valor práctico del modelo, una posible extensión del trabajo sería simular un sistema de apuestas deportivas basado en las predicciones obtenidas. Esto permitiría comprobar hasta qué punto el modelo podría ser rentable en escenarios reales. Una opción sería utilizar un programa de optimización o incluso diseñar un agente de *Reinforcement Learning* que, en función de las probabilidades predichas para cada clase, decidiera cuánto apostar y a qué resultado. La recompensa de este agente sería el beneficio obtenido. Sin embargo, este enfoque requeriría una implementación mucho más compleja y una expansión considerable del alcance del trabajo, por lo que se ha descartado en esta fase.

5.5 Conclusión Personal

Este trabajo ha supuesto una forma muy especial de cerrar la etapa del grado universitario. A lo largo de la carrera, la inteligencia artificial y el *Machine Learning* han sido algunos de los temas que más interés y motivación me han generado, tanto por su potencial como por su aplicabilidad en problemas reales.

Desarrollar este proyecto me ha permitido poner en práctica, de forma integral, muchas de las habilidades adquiridas durante los estudios. Desde el análisis y preprocesamiento de los datos hasta la implementación, ajuste y evaluación de modelos, cada fase ha representado un reto y una oportunidad de aprendizaje. Considero que este TFG ha sido la mejor manera de aplicar de forma concreta todo lo aprendido sobre aprendizaje automático y, al mismo tiempo, explorar en mayor profundidad un área que me resulta especialmente estimulante y en la que me gustaría seguir profundizando en el futuro.

6. REFERENCIAS

Todas las imágenes, citas al pie y referencias del TFG quedan registradas en este apartado siguiendo la normativa APA7.

6.1 Bibliografía

[1] KPMG Asesores S.L. (2023). Impacto socioeconómico del fútbol profesional en España. KPMG.

[2] Aoki, R. Assunção, R. Vaz de Melo, P. (2017) Luck is Hard to Beat: The Dificulty of Sports Prediction. Department of Computer Science UFMG.

[3] Cavus, M. Biecek, P. (2022). Explainable Expected Goal Models for Performance Analysis in Football Analytics. Warsaw University of Technology. Eskisehir Technical University.

[4] Spearman, W. (2028). Beyond Expected Goals. Sports Analytics Conference. MIT Sloan, Boston MA.

[5] Manassé Galekwa, R. Tshimula, J. M. Tajeuna, E. G. Kyandoghere, K. Systematic Review of Machine Learning in Sports Betting: Techniques, Challenges, and Future Directions.

[6] Lora Gutiérrez, J. D., & Macias Burbano, Y. A. (2022). Ineficiencias en el mercado de apuestas del fútbol: Los efectos del COVID-19 [Tesis de grado, Universidad Icesi].

[7] Van Raaij, V. (2019). Favorite-longshot bias in European Football betting market: Differences between popular and non-popular football competitions. Radboud Universiteit.

[8] Martín Domínguez, D. (2013). Análisis de resultados deportivos y estimación implícita de probabilidades: Fútbol. UC3M

[9] A. Jung,“Machine Learning: The Basics,” Springer, Singapore, 2022.

[10] Neupert, T., Fischer, M. H., Greplova, E., Choo, K., & Denner, M. M. (2021). Introduction to Machine Learning for the Sciences.

[11] Domingos, P. (2012). A few useful things to know about machine learning. Communications of the ACM.

[12] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering.

[13] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied Logistic Regression (Vol. 398). John Wiley & Sons.

[14] Cunningham, P. Delany, S.J. (2020). K-Nearest Neighbour Classifiers 2nd Edition (with Python examples). University College Dublin. Technological University Dublin

[15] Izza, Y. Ignatiev, A. Marques-Silva, J. ANITI, (2020). On Explaining Decision Trees. Univ. Toulouse

[16] Louppe, G. (2015). Understanding Random Forests. University of Liège

[17] He, T., Zhao, H., Chu, X., Liu, Z., Liu, X., & Chen, G. (2019). Gradient Boosting: A Survey. Point Zero One Technology. Fordham University. Imperial College London. Massachusetts Institute of Technology.

[18] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting System.

[19] Qamar, R., Zardari, B. A. (2023). Artificial Neural Networks: An Overview. Vol. (2023). Mesopotamian Journal of Computer Science

[20] Herbinet, C. (2018). Using machine learning techniques to predict the outcome of professional football matches [Undergraduate project report, Imperial College London]. Imperial College London Repository.

[21] Soto-Valero, C. (2018). Aplicación de métodos de aprendizaje automático en el análisis y la predicción de resultados deportivos. Retos: Nuevas Tendencias en Educación Física, Deporte y Recreación.

[22] Martínez Arias, L. M., & Marulanda Vélez, S. (2023). Modelo de clasificación multiclases para la predicción de apuestas deportivas [Trabajo de grado especialización, Universidad de Antioquia]. Repositorio Institucional Universidad de Antioquia.

[23] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

[24] Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research

[25] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets.

[26] Baboota, R. & Kaur, H. (2019). Predictive analysis and modelling football results using machine learning approach for English Premier League. International Journal of Forecasting

[27] Alfredo, Y., et al. (2020). Football match prediction using tree-based models. Proc. of STI 2020

[28] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE.

6.2 Índices de Imágenes y Figuras

6.2.1 Imágenes

[1] Captura de pantalla. Aplicación Móvil. Plataforma de resultados Flashscore: Apuestas 1X2

[2] Captura de pantalla. Aplicación Móvil. Plataforma de resultados Flashscore: Hándicap Asiático

[3] Bagnato, J. I. (2017, 12 diciembre). Representación gráfica de los conceptos de overfitting y underfitting. Aprende Machine Learning. https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/

[4] Izco, F. (s.f.). Matriz de confusión. BDC-POC. https://bookdown.org/f_izco/BDC-POC/metricas.html

[5] Díaz Sosa, M. L. (s.f.). Representación de la función sigmoide. GeoGebra. https://www.geogebra.org/m/MpJeZcMB

[6] Qamar, R., & Zardari, B. A. (2023). Artificial Neural Networks: An Overview. Mesopotamian Journal of Computer Science, 2023. https://doi.org/10.58496/MJCSC/2023/015

[7] Interfaz de MLflow en Dagshub. Captura de pantalla.

https://dagshub.com/anaigs/tfg_inso_github

[8] Grafico de rendimiento que produce MLflow al comparar modelos. Captura de pantalla.

https://dagshub.com/anaigs/tfg_inso_github

[9] Dataset completo de partidos desde la temporada 03/04 hasta 23/24. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/2-analisis_exploratorio.ipynb

[10] Distribución de goles por partido. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/2-analisis_exploratorio.ipynb

[11] Rendimiento en casa vs. fuera . Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/2-analisis_exploratorio.ipynb

[12] Distribución de cambios de posición entre temporadas en LaLiga. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/2-analisis_exploratorio.ipynb

[13] Ejemplo de los últimos cinco partidos del Real Madrid en el dataset. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/3-generacion_dartasets.ipynb

[14] Evolución del Rendimiento por Temporadas. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/4-analisis_datasets.ipynb

[15] Distribución de la Diferencia de Goles por Equipo. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/4-analisis_datasets.ipynb

[16] Porcentaje de Victorias en Casa y Fuera por Equipo. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/4-analisis_datasets.ipynb

[17] Relación entre Cuotas y Resultados. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/data_cleaning/4-analisis_datasets.ipynb

[18] Comparativa de f1_score entre Experimentos de Regresión Logística.

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

[19] Matriz de Confusión del Modelo Básico para el Real Madrid. Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/models /modelos_lineales/regresion_logistica.ipynb

[20] Comparativa de Configuraciones y F1-score para Athletic Club de Bilbao.

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

[21] Comparativa de F1-score entre distintos enfoques de KNN.

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

[22] Matriz de confusión del modelo KNN optimizado (Athletic Club). Generado en:

Repositorio: https://github.com/anaigs/tfg_inso_github

Ruta: code/models /modelos_no_lineales /KNN.ipynb

[23] Comparativa de F1-score en modelos Random Forest.

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

[24] Comparativa entre Modelos Random Forest (Real Madrid).

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

[25] Comparativa de F1-score entre modelos Boosting.

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

[26] Comparativa de F1-score entre modelos con redes neuronales.

Captura de pantalla. https://dagshub.com/anaigs/tfg_inso_github

6.2.2 Figuras

[1] Tabla de rendimiento. Mejores modelos por equipo según F1-score, considerando todas las configuraciones evaluadas en el estudio.

Datos sacados de https://dagshub.com/anaigs/tfg_inso_github

[2] Tabla de rendimiento. Comparativa de los cuatro mejores modelos para el equipo Real Madrid, ordenados por F1-score. Se incluyen las técnicas aplicadas y el tamaño del conjunto de prueba.

[3] Tabla de rendimiento. Comparativa de los cuatro mejores modelos para el equipo FC Barcelona, ordenados por F1-score. Se detallan las técnicas utilizadas y el tamaño del conjunto de prueba.

[4] Tabla de rendimiento. Comparativa de los cuatro mejores modelos para el equipo Athletic de Bilbao, ordenados por F1-score. Se indican las técnicas aplicadas y el tamaño del conjunto de prueba.

[5] Tabla de rendimiento. Comparativa de los cuatro mejores modelos para el equipo Valencia CF, ordenados por F1-score. Se especifican las técnicas utilizadas y el tamaño del conjunto de prueba.

ANEXOS

Anexo A. Registro de experimentos en Mlflow (Random Forest)

Este fragmento de código muestra cómo se registra un experimento en MLflow, herramienta empleada a lo largo del proyecto para el seguimiento y gestión de modelos entrenados. En este ejemplo se registra un modelo de Random Forest aplicado al Athletic Club, tras una validación secuencial con optimización de hiperparámetros.

El entrenamiento y la validación del modelo pueden realizarse también dentro del bloque *mlflow.start_run()*, pero en este caso se han ejecutado previamente para que el ejemplo se centre únicamente en cómo registrar los parámetros, métricas y artefactos del experimento.

```python
with mlflow.start_run(run_name="athletic_bilbao_rf_optimized"):
    mlflow.set_tag("team", "athletic")
    mlflow.log_param("model_type", "Random Forest")
    mlflow.log_param("approach", "Sequential Validation + Hyperparameter Tuning")
    mlflow.log_param("team", "Athletic Bilbao")
    mlflow.log_param("test_size", n_validacion)

    # Registro de hiperparámetros
    mlflow.log_param("n_estimators", best_params["n_estimators"])
    mlflow.log_param("max_depth", best_params["max_depth"])
    mlflow.log_param("min_samples_split", best_params["min_samples_split"])
    mlflow.log_param("min_samples_leaf", best_params["min_samples_leaf"])

    # Registro de métricas obtenidas tras la validación
    mlflow.log_metric("accuracy", accuracy)
    mlflow.log_metric("precision", precision)
    mlflow.log_metric("recall", recall)
    mlflow.log_metric("f1_score", f1)

    # Guardar y subir como artefacto
    matrix_path = "conf_matrix_rf_athletic.png"
    mlflow.log_artifact()
    os.remove(matrix_path)
```

Anexo B. Ejemplo de Validación Secuencial Acumulativa (Regresión Logística)

Este fragmento muestra cómo se lleva a cabo la validación secuencial acumulativa, técnica utilizada a lo largo del estudio para simular la incorporación progresiva de nuevos partidos al modelo. En este ejemplo, se aplica con un modelo de regresión logística.

```python
y_pred_seq = []
y_real_seq = []
# Validación secuencial acumulativa
for i in range(n_validacion):
    # Entrenar el modelo con todos los partidos anteriores
    model = LogisticRegression(solver=solver, max_iter=max_iter, random_state=42)
    model.fit(X_train_scaled, y_train_init)

    # Obtener el partido a predecir
    X_next = X_val_seq.iloc[i:i+1]
    y_next = y_val_seq.iloc[i]

    X_next_scaled = scaler.transform(X_next) # Escalar con los mismos parámetros del entrenamiento
    y_pred_next = model.predict(X_next_scaled)[0] # Predecir el partido

    # Guardar la predicción y el resultado real
    y_pred_seq.append(y_pred_next)
    y_real_seq.append(y_next)

    # Agregar el partido actual al conjunto de entrenamiento para la siguiente iteración
    X_train_init = pd.concat([X_train_init, X_next])
    y_train_init = pd.concat([y_train_init, pd.Series([y_next])])

    # Volver a escalar los datos con la nueva información incluida
    X_train_scaled = scaler.fit_transform(X_train_init) # Volver a escalar los datos
```

Anexo C. Optimización de hiperparámetros con GridSearchCV

Este fragmento muestra cómo se utiliza GridSearchCV para buscar la mejor combinación de hiperparámetros en un modelo de Random Forest. El proceso consiste en definir un conjunto de valores candidatos para cada parámetro y aplicar validación cruzada para seleccionar la

mejor configuración. Aunque este ejemplo está centrado en Random Forest, la estructura es la misma para otros modelos como KNN, o Gradient Boosting, modificando únicamente los parámetros a evaluar.

```python
# Definir los hiperparámetros a optimizar
param_grid = {
    "n_estimators": [50, 100, 200],
    "max_depth": [None, 10, 20, 30],
    "min_samples_split": [2, 5, 10],
    "min_samples_leaf": [1, 2, 4]
}
# Aplicar GridSearchCV con validación cruzada
grid_search = GridSearchCV(
    RandomForestClassifier(random_state=42),
    param_grid,
    cv=5,
    scoring="accuracy",
    n_jobs=-1
)
grid_search.fit(X_train_scaled, y_train_init)
# Obtener los mejores parámetros encontrados
best_params = grid_search.best_params_
```

Anexo D. Optimización de Hiperparámetros con Optuna (Ejemplo en XGBoost)

El siguiente fragmento de código muestra cómo se ha utilizado la biblioteca Optuna para llevar a cabo una optimización eficiente de hiperparámetros en el modelo XGBoost, mediante validación interna con train_test_split:

```python
def objective(trial):
    params = {
        "n_estimators": trial.suggest_int("n_estimators", 50, 300),
        "max_depth": trial.suggest_int("max_depth", 3, 15),
        "learning_rate": trial.suggest_float("learning_rate", 0.01, 0.3),
        "min_child_weight": trial.suggest_int("min_child_weight", 1, 10),
        "subsample": trial.suggest_float("subsample", 0.5, 1.0),
        "colsample_bytree": trial.suggest_float("colsample_bytree", 0.5, 1.0),
        "gamma": trial.suggest_float("gamma", 0, 5),
        "lambda": trial.suggest_float("lambda", 0, 5)
    }
    # División del conjunto para validación interna
    X_train_opt, X_val_opt, y_train_opt, y_val_opt = train_test_split(
        X_train_pca, y_train, test_size=0.2, random_state=42, stratify=y_train
    )
    # Entrenar el modelo con los parámetros actuales
    model = xgb.XGBClassifier(
        **params, random_state=42, use_label_encoder=False, eval_metric="mlogloss"
    )
    model.fit(X_train_opt, y_train_opt)

    y_pred_opt = model.predict(X_val_opt) # Evaluación
    return accuracy_score(y_val_opt, y_pred_opt)

# Ejecutar la optimización con Optuna
study = optuna.create_study(direction="maximize")
study.optimize(objective, n_trials=100)

# Obtener los mejores parámetros
best_params = study.best_params
```