Universidad de Valladolid

Facultad de Ciencias

TRABAJO FIN DE GRADO

Grado En Estadística

Predicción de resultados en partidos de tenis mediante web scraping y redes neuronales

Autor:
Diego Rodríguez de Roa

Tutores:
Diego Rafael Llanos Ferraris
Francisco Hernando Gallego

Curso 2024-2025

Agradecimientos

En primer lugar, quiero expresar mi más sincero agradecimiento a mis tutores del trabajo, D. Francisco Hernando y D. Diego R. Llanos, cuyo apoyo y conocimiento técnico fueron fundamentales para el desarrollo de este proyecto. Incluso cuando planteaba consultas urgentes, siempre conté con comprensión y orientación hacia soluciones acertadas, mostrando en todo momento una gran disposición a ayudarme.

Agradecer también a los profesores del Grado en Estadística de la Universidad de Valladolid por proporcionarme los conocimientos teóricos y las herramientas prácticas necesarias para abordar este proyecto. Su formación me ha proporcionado las competencias necesarias que ahora comienzo a poner en práctica en esta nueva etapa profesional que se inicia para mí.

También quiero expresar mi reconocimiento a mis compañeros de estudios, con quienes he compartido experiencias valiosas a lo largo de estos cinco años de formación. Gracias a todos ellos por la compañía y las vivencias compartidas.

Por último, quiero agradecer especialmente a mis padres por brindarme la educación y los valores que me han guiado hasta este momento. Su apoyo incondicional, tanto a lo largo de mi formación como durante la realización de este trabajo de fin de grado, ha sido fundamental para que hoy pueda poner fin a esta etapa.

Resumen

La predicción de resultados en partidos de tenis representa uno de los desafíos más complejos del análisis deportivo, donde la incertidumbre inherente al deporte se combina con múltiples factores que determinan el desenlace final. Este proyecto desarrolla un sistema que integra información histórica de rendimiento con características contextuales de cada encuentro mediante redes neuronales recurrentes, procesando datos del circuito ATP profesional del período 2021-2024 a través de un sistema automatizado de extracción que integra múltiples fuentes. Se desarrollan dos modelos complementarios: uno de clasificación para predecir el resultado binario y otro de regresión para estimar probabilidades de victoria ofrecidas por las casas de apuestas.

Palabras Clave: Predicción deportiva, Redes neuronales recurrentes, Tenis profesional

Abstract

Tennis match outcome prediction represents one of the most complex challenges in sports analytics, where the inherent uncertainty of the sport combines with multiple factors that determine the final result. This project develops a system that integrates historical performance data with contextual characteristics of each match through recurrent neural networks, processing ATP professional circuit data from the 2021-2024 period through an automated extraction system that integrates multiple sources. Two complementary models are developed: a classification model to predict binary outcomes and a regression model to estimate victory probabilities offered by bookmakers.

Keywords: Sports prediction, Recurrent neural networks, Professional tennis.

Índice general

Lista de Figuras

Lista de Tablas

Glosario de términos

A | C | E | G | H | J | M | P | S

A

API

Application Programming Interface. Interfaz que permite la comunicación entre sistemas software mediante el protocolo HTTP. Facilita el acceso a datos en formato JSON a través de URLs específicas. 13, 14, 16, 18, 50, XVII

ATP

Association of Tennis Professionals. Organización que gobierna el circuito profesional masculino de tenis a nivel mundial. Establece el sistema de clasificación oficial mediante puntos ATP obtenidos en torneos profesionales. El circuito ATP comprende las categorías principales 2, 9, 14, 16–22, 25, 26, 49, 50, XVII

ATP 500

Categoría de torneos del circuito ATP que otorgan 500 puntos al campeón y se disputan al mejor de 3 sets. Representan el tercer nivel competitivo tras los Grand Slams y Masters 1000 14, 17, 22, XVII

ATP Finals

Torneo de fin de temporada que reúne a los ocho mejores jugadores del ranking ATP. Se disputa al mejor de 3 sets, otorgando hasta 1500 puntos al campeón. 17, 22, XVII

C

CRISP-DM

CRoss-Industry Standard Process for Data Mining, metodología utilizada para proyectos de minería de datos que consta de seis fases: comprensión del negocio, comprensión de los datos, preparación de los datos, modelado, evaluación y despliegue. 2–4, XI, XVII

E

Endpoint

URLs que proporcionan acceso a recursos específicos dentro de una API 14–19, 21, XVII

G

Grand Slam

Los cuatro torneos de tenis más prestigiosos del mundo: Australian Open, Roland Garros (French Open), Wimbledon y US Open. Estos eventos representan el máximo nivel competitivo en el tenis profesional, distinguiéndose por otorgar la mayor cantidad de puntos de ranking ATP (2000 puntos al campeón) y por ser los únicos torneos del circuito masculino que se disputan al mejor de 5 sets. 11, 14, 17, 22, 25, XVII

H

HTML

HyperText Markup Language. Lenguaje de marcado que define la estructura y contenido de páginas web mediante etiquetas y elementos. Su estructura anidada complejiza la extracción automatizada de datos. 13, 14, XVII

J

JSON

Formato de intercambio de datos basado en texto, usado comúnmente en APIs para estructurar datos en pares clave-valor. 13, 14, 16, XI, XVII

M

Masters 1000

Serie de nueve torneos anuales que constituyen el segundo nivel más alto del circuito ATP, otorgando 1000 puntos al campeón y disputándose al mejor de 3 sets. Incluye eventos como Indian Wells, Miami, Monte Carlo y Madrid 14, 17, 22, XVII

Matchstat

Plataforma specializada en estadísticas de tenis. Proporciona información histórica sobre jugadores, enfrentamientos directos, rankings y resultados de torneos. 18–20, XI, XVII

P

Pandas

Biblioteca de Python para el análisis y manipulación de datos estructurados. Facilita operaciones de limpieza, transformación y análisis de datos, y permite importar y exportar información en múltiples formatos. 16, XVII

Python

Lenguaje de programación interpretado, destaca por su sintaxis intuitiva y su ecosistema de librerías para análisis de datos (pandas, numpy), visualización (matplotlib), aprendizaje automático (scikit-learn, Pytorch) y web scraping (requests) 16, XVII

S

Sofascore

Plataforma de eventos deportivos que ofrece estadísticas detalladas de múltiples deportes, incluyendo tenis. Obtiene los datos a través de una API estructurada. 11–13, 16–21, 24, 50, XI, XVII

Capítulo 1

Introducción

1.1. Contexto y Motivación

La predicción de resultados en eventos deportivos representa uno de los desafíos más complejos dentro del análisis predictivo. En deportes como el tenis, la incertidumbre propia de cada encuentro, combinada con la influencia de múltiples factores como el rendimiento individual, las condiciones físicas y el contexto del torneo, convierte la predicción del resultado en una tarea de alta complejidad. Esta naturaleza dinámica convierte al deporte profesional en un campo especialmente adecuado para la aplicación de técnicas de aprendizaje automático, capaces de extraer patrones significativos de grandes volúmenes de información histórica y contextual. Sin embargo, la disponibilidad de estos datos requiere la implementación de técnicas de web scraping que permitan extraer información de manera automatizada desde plataformas deportivas especializadas, constituyendo un paso fundamental para el desarrollo de modelos predictivos efectivos.

El tenis profesional presenta características que lo convierten en un caso de estudio particularmente interesante para el desarrollo de modelos predictivos. Al tratarse de un deporte individual, cada partido se reduce a un enfrentamiento directo entre dos jugadores, lo que facilita la identificación y el análisis de las variables específicas que influyen en el resultado final. No obstante, esta aparente simplicidad no reduce la complejidad predictiva, ya que factores como el tipo de superficie de juego, el historial de enfrentamientos previos y el ranking de cada jugador interactúan de manera compleja para determinar el desenlace final.

Desde una perspectiva comercial y práctica, el desarrollo de herramientas predictivas precisas resulta relevante en múltiples ámbitos deportivos. La industria de apuestas deportivas constituye uno de los sectores que mayor beneficio podría obtener de estos sistemas, utilizando modelos capaces de predecir el resultado de los encuentros.

La motivación de este proyecto surge de la oportunidad de aplicar técnicas de aprendizaje automático a un problema real, donde la integración de información temporal y contextual

puede ofrecer ventajas predictivas significativas más allá del uso exclusivo de variables propias del partido a predecir. El tenis profesional constituye un escenario ideal para explorar arquitecturas que combinen el procesamiento de información histórica con el análisis de características contextuales del encuentro actual.

1.2. Objetivos del proyecto

El objetivo principal de este trabajo consiste en desarrollar un sistema de predicción de resultados para partidos de tenis del circuito ATP profesional, empleando técnicas de aprendizaje automático que integren información histórica de rendimiento con características contextuales de cada encuentro.

Para alcanzar este propósito general, se han establecido objetivos específicos que abordan tanto aspectos técnicos como de aplicabilidad práctica. En primer lugar, se busca automatizar el proceso de extracción de datos mediante la implementación de un sistema de web scraping. Este proceso se define como la extracción y combinación de información de interés desde la Web de manera sistemática, capaz de recopilar información desde múltiples fuentes [1]. Esta técnica debe garantizar la integridad temporal de los datos y asegurar la coherencia entre diferentes tipos de información, logrando una integración efectiva que mantenga la calidad de los datos obtenidos.

También se busca el desarrollo de un modelo predictivo de aprendizaje automático que muestre precisión y estabilidad a lo largo del tiempo. Además, este modelo debe ser capaz de procesar tanto información histórica secuencial como características contextuales del encuentro actual. El sistema debe mantener su efectividad predictiva de manera continuada en el tiempo.

Finalmente, se busca la implementación de un sistema con aplicabilidad comercial, desarrollando una herramienta capaz de generar valor en contextos comerciales reales, especialmente en sectores como casas de apuestas o plataformas de análisis deportivo. El sistema debe identificar oportunidades en el mercado y proporcionar predicciones que faciliten la toma de decisiones en entornos de predicción deportiva

Estos objetivos se orientan hacia la creación de una solución completa que demuestre la efectividad de las redes neuronales recurrentes aplicadas en el contexto deportivo. A su vez, se busca sentar las bases para su implementación práctica en entornos comerciales, donde una mayor precisión predictiva puede convertirse en una oportunidad de valor real.

1.3. Metodología empleada

La metodología empleada en este trabajo sigue un enfoque adaptado de CRISP-DM (Cross-Industry Standard Process for Data Mining), cuya estructura sistemática resulta adecuada para proyectos basados en técnicas de aprendizaje automático. Esta metodología se

estructura en seis fases principales que definen un ciclo de vida iterativo para los proyectos, como se ilustra en la figura 1.1. Según Chapman et al. (2000) [2], las fases aplicadas al contexto de este trabajo son:

- **Comprensión del problema:** Definición del problema de predicción en tenis profesional, estableciendo los objetivos a lograr.
- **Comprensión de los datos:** Identificación y análisis de las fuentes de información disponibles, incluyendo la exploración de las características del conjunto de datos extraído.
- **Preparación de los datos:** Implementación de un sistema de web scraping para extracción automatizada de los datos deportivos. Tras ello, se aplican las técnicas de preprocesamiento, como la transformación de variables y la estructuración de los datos para su uso en modelos de aprendizaje automático.
- **Modelado:** Diseño e implementación de los modelos de redes neuronales.
- **Evaluación:** Análisis del rendimiento de los modelos implementados, comparando los resultados obtenidos en ellos.
- **Aplicación práctica:** Propuesta de implementación del sistema de extracción y de predicción en un escenario real.

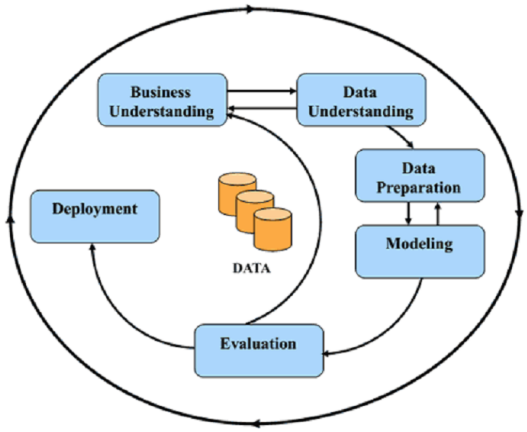

Figura 1.1: Fases de la metodología CRISP-DM [3].

Esta metodología proporciona un marco estructurado que garantiza un desarrollo adecuado del proyecto, desde la concepción inicial hasta la evaluación de su aplicabilidad práctica en entornos comerciales.

1.4. Estructura de la memoria

Este documento se organiza siguiendo la metodología CRISP-DM presentada anteriormente, donde cada capítulo corresponde a una fase específica del proceso. Esta estructura garantiza una progresión coherente desde la identificación del problema hasta la evaluación de la solución propuesta, respetando las fases de metodología a lo largo del proceso:

Capítulo 1 - Introducción: Corresponde a la fase de comprensión del problema, estableciendo el contexto del proyecto, la motivación que lo impulsa, los objetivos específicos a alcanzar y la metodología empleada para su desarrollo.

Capítulo 2 - Justificación teórica y estado del arte: Complementa la fase de comprensión del problema mediante la presentación del fundamento teórico de las redes neuronales recurrentes y el análisis de trabajos previos en el ámbito de la predicción deportiva y las técnicas de aprendizaje automático aplicadas al tenis.

Capítulo 3 - Los datos: Abarca las fases de comprensión y preparación de los datos. En la primera parte se presentan las fuentes de información utilizadas, la implementación del sistema de web scraping para la extracción automatizada de información, y el análisis descriptivo del conjunto de datos obtenido. La segunda parte detalla las técnicas de preprocesamiento aplicadas, incluyendo la normalización, transformación de variables y estructuración final de los DataSets.

Capítulo 4 - El modelo: Corresponde a la fase de modelado, donde se presenta la arquitectura híbrida desarrollada y la implementación de los modelos de clasificación y regresión. Se incluyen los detalles técnicos del entrenamiento y validación.

Capítulo 5 - Resultados: Engloba las fases de evaluación y aplicación práctica. Se presenta el análisis del rendimiento de ambos modelos, la comparación de resultados obtenidos, y la evaluación de la viabilidad comercial, incluyendo propuestas de implementación en escenarios reales.

Capítulo 6 - Conclusiones: Sintetiza los logros alcanzados, las limitaciones identificadas y las líneas de trabajo futuro, proporcionando una evaluación completa del proyecto.

Capítulo 2

Justificación teórica y estado del arte

Este capítulo establece las bases teóricas necesarias para comprender el modelo predictivo desarrollado, presentando los fundamentos de las redes neuronales recurrentes y analizando el estado de la investigación en predicción deportiva aplicada al tenis profesional. Además, se justifica la elección del enfoque propuesto mediante un análisis que identifica las limitaciones de los métodos existentes y las ventajas de la arquitectura híbrida implementada.

2.1. Fundamento teórico

2.1.1. Aprendizaje automático y Deep Learning

La capacidad de procesar y extraer patrones de grandes volúmenes de datos constituye uno de los desafíos fundamentales de la inteligencia artificial, entendida como la disciplina que busca desarrollar sistemas computacionales capaces de realizar tareas que tradicionalmente requieren inteligencia humana [4]. En las últimas décadas, el desarrollo de técnicas de Deep Learning ha permitido abordar problemas analíticos que anteriormente parecían intratables, particularmente en problemas donde las relaciones entre variables presentan una alta complejidad.

Dentro de la inteligencia artificial, el aprendizaje automático se centra en desarrollar sistemas capaces de extraer conocimiento de los datos y mejorar su rendimiento con la experiencia. El Deep Learning constituye un subconjunto especializado de estas técnicas, caracterizado por el uso de redes neuronales con múltiples capas que logran descubrir de manera automática las relaciones subyacentes en los datos [5].

Las redes neuronales artificiales representan modelos computacionales inspirados en el

funcionamiento del cerebro humano. Sus fundamentos teóricos se establecieron en 1943 cuando McCulloch y Pitts desarrollaron el primer modelo matemático de neuronas artificiales [6]. Estas arquitecturas poseen la capacidad de aprender representaciones complejas de forma automática, extrayendo características relevantes sin requerir procesamiento manual de variables. Dicha capacidad supone una ventaja significativa frente a los enfoques tradicionales, ya que permite que los sistemas identifiquen patrones complejos de manera autónoma [7].

Las redes feedforward, también conocidas como perceptrones multicapa (MLP), representan la arquitectura más básica. Estas redes organizan las neuronas en capas secuenciales con flujo unidireccional en tres capas, capa de entrada, capas ocultas y capa de salida [8]. Sin embargo, esta arquitectura presenta limitaciones significativas para datos secuenciales, ya que procesan cada entrada de manera independiente, sin capacidad de mantener información sobre estados anteriores o establecer dependencias temporales.

2.1.2. Redes Neuronales recurrentes

Las redes neuronales recurrentes constituyen una arquitectura diseñada para procesar datos secuenciales mediante un estado oculto que actúa como memoria de la red [9]. A diferencia de las redes feedforward, las RNN incorporan conexiones recurrentes que permiten considerar tanto la entrada actual como las entradas anteriores, manteniendo así información temporal. El funcionamiento de una RNN se describe mediante las siguientes ecuaciones [10]:

$$h_t = \sigma_h \left(W_{xh} x_t + W_{hh} h_{t-1} + b_h \right)$$
$$y_t = \sigma_y \left(W_{hy} h_t + b_y \right)$$

donde h_t representa el estado oculto en el instante t, x_t es la entrada actual, y_t constituye la predicción actual del modelo, y las matrices W son los pesos entrenables. El estado oculto incluye recursivamente información de todos los estados anteriores, permitiendo que la red mantenga memoria de la secuencia temporal de información [9]. La figura 2.1 muestra la estructura básica de una red neuronal recurrente.

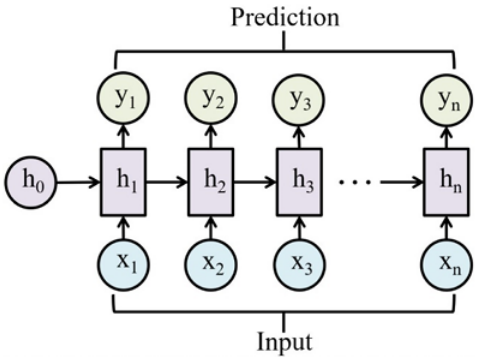

Figura 2.1: Estructura básica de una RNN para el instante t, donde x_t representa la entrada, h_t el estado oculto e y_t la predicción actual. Fuente: [9]

El entrenamiento de las RNN requiere una adaptación del algoritmo de retropropagación tradicional (utilizado para calcular los gradientes de error y ajustar los pesos en redes feedforward) conocida como Backpropagation Through Time (BPTT) [10]. Este algoritmo calcula gradientes aplicando la regla de la cadena a través de múltiples pasos temporales, resultando en expresiones que involucran productos de matrices Jacobianas.

Durante este proceso, los gradientes pueden experimentar problemas de estabilidad: si los valores propios de las matrices de pesos son menores que 1, los gradientes tienden a cero conforme aumenta la longitud de la secuencia (gradiente desvaneciente). Por otra parte, si son mayores que 1, los gradientes crecen exponencialmente hacia infinito (gradiente explosivo), causando en ambos casos inestabilidad numérica y dificultades de convergencia. Este hecho es muy problemático, ya que impide el aprendizaje efectivo de dependencias a largo plazo al limitar la contribución de los estados iniciales hacia la predicción actual [10].

2.1.3. Soluciones Avanzadas: LSTM y GRU

Los problemas de gradiente identificados en las RNN tradicionales motivaron el desarrollo de arquitecturas más sofisticadas que incorporan mecanismos de compuertas para controlar el flujo de información. Las dos soluciones principales son las Long Short-Term Memory (LSTM) y Gated Recurrent Units (GRU) [9].

Las redes LSTM abordan el problema del gradiente desvaneciente mediante tres compuertas especializadas: la compuerta de entrada, la compuerta de olvido y la compuerta de salida [11]. Estas compuertas regulan el estado de la celda y el estado oculto, permitiendo que la red actualice selectivamente su memoria interna durante períodos prolongados, facilitando así el aprendizaje de dependencias temporales a largo plazo. La figura 2.2 muestra la estructura de una celda LSTM. Por su parte, las GRU constituyen una variante simplificada que combina

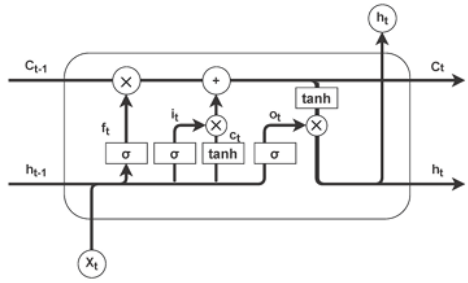

Figura 2.2: Estructura básica de una celda LSTM para el instante t, donde i_t, f_t y o_t representan las compuertas de entrada, olvido y salida respectivamente, c_t es el estado de la celda, h_t el estado oculto y x_t la entrada candidata de la celda. Fuente: [9]

las compuertas de entrada y olvido en una sola compuerta de actualización, además de introducir una compuerta de reinicio [12]. Esta arquitectura reduce significativamente el número de parámetros mientras se mantiene un rendimiento comparable al de las LSTM en muchas

tareas, resultando en una mayor eficiencia computacional y tiempos de entrenamiento más rápidos. La figura 2.3 presenta la estructura de una celda GRU.

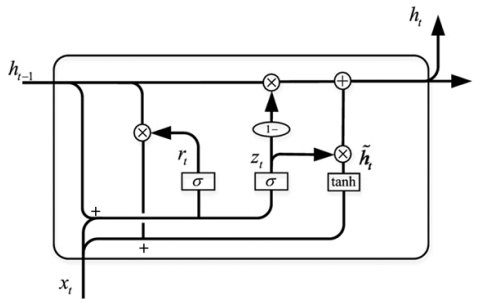

Figura 2.3: Estructura básica de una celda GRU para el instante t, donde z_t representa la compuerta de actualización, r_t la compuerta de reinicio, h_t el estado oculto y h'_t el estado oculto candidato. Fuente: [9]

La elección entre LSTM y GRU depende del equilibrio entre precisión y eficiencia computacional. Mientras que las LSTM pueden ofrecer ventajas en tareas que requieren modelar dependencias muy complejas, las GRU proporcionan una alternativa eficiente especialmente adecuada cuando se requiere entrenamiento más rápido [9].

2.2. Estado del arte

2.2.1. Predicción de resultados en el tenis profesional

La predicción de resultados deportivos ha evolucionado desde enfoques estadísticos tradicionales hasta técnicas avanzadas de aprendizaje automático. El tenis profesional presenta características que lo convierten en un deporte particularmente interesante para el desarrollo de modelos predictivos. La naturaleza jerárquica del tenis, donde el resultado final depende de la acumulación de puntos, juegos y sets, ofrece múltiples niveles de análisis para el desarrollo de modelos predictivos.

Los enfoques predictivos han variado con el tiempo desde modelos estadísticos basados en supuestos de independencia entre puntos hasta sistemas más sofisticados que incorporan información contextual y temporal. Sin embargo, la literatura existente revela limitaciones importantes en el tratamiento de la información temporal y en la estabilidad de las predicciones a lo largo del tiempo.

2.2.2. Modelos estadísticos tradicionales

Los modelos estadísticos tradicionales para la predicción de resultados en partidos de tenis se fundamentan en dos enfoques principales: los modelos basados en puntos y las expresiones

recursivas. Sipko y Knottenbelt (2015) [13] desarrollaron modelos basados en cadenas de Markov asumiendo que los puntos de un partido son independientes e igualmente distribuidos. Estos modelos se centran en estimar la probabilidad de que un jugador gane un punto con su servicio, incorporando estadísticas como el porcentaje de primeros servicios, los puntos ganados en segundo servicio y el rendimiento al devolver el servicio.

Las expresiones recursivas constituyen la extensión natural de este enfoque, permitiendo modelar situaciones más complejas mediante definiciones matemáticas que relacionan la probabilidad de ganar un juego con el estado actual del marcador en el partido [13].

De manera complementaria, De Seranno (2020) [14] implementó modelos Bradley-Terry que utilizan el método de máxima verosimilitud sobre partidos históricos para calcular estas probabilidades fundamentales, demostrando la versatilidad de estos enfoques estadísticos.

Estos modelos se basan en supuestos que han sido objeto de análisis crítico. De Seranno (2020) [14] señala que, aunque estudios posteriores demostraron que la hipótesis de independencia entre puntos no se cumple estrictamente, las desviaciones observadas resultan lo suficientemente pequeñas como para que la asunción de probabilidad constante durante el partido siga siendo adecuada para propósitos predictivos. No obstante, la principal limitación de estos enfoques radica en su capacidad limitada para incorporar información contextual más allá de las estadísticas básicas de servicio.

2.2.3. Modelos de aprendizaje automático

Los enfoques de aprendizaje automático muestran capacidades superiores a los modelos estadísticos tradicionales al incorporar mayor número de variables contextuales y capturar relaciones más complejas. Sipko y Knottenbelt (2015) [13] exploraron máquinas de soporte vectorial (SVM) que buscan hiperplanos de margen máximo para separar las clases ganador y perdedor, aunque con resultados limitados debido a la complejidad de las relaciones entre variables.

Por su parte, De Seranno (2020) [14] implementó un modelo de regresión logística con 38 variables contextuales, incluyendo tipo de superficie, edad de los jugadores y formato del partido, alcanzando una precisión del 67.4 % que superó al 64.5 % obtenido por un modelo basado únicamente en el ranking ATP. Este resultado evidencia la importancia de incorporar información contextual más allá de las clasificaciones.

Fang et al. (2024) [15] extendieron estos enfoques mediante Random Forest para la predicción de sets, combinando múltiples árboles de decisión entrenados con técnicas de bagging y utilizando métricas específicas como el número de aces y los puntos consecutivos ganados, logrando una precisión del 79.17 % en la predicción de sets individuales.

Por otra parte, las redes neuronales han mostrado un alto potencial para capturar patrones complejos en datos de partidos de tenis, especialmente cuando se adaptan a los distintos niveles en los que se puede analizar un encuentro, como puntos, juegos o sets. De Seranno (2020) [14] implementó redes neuronales “feed forward” con 84 características de entrada, alcanzando una precisión del 68.2 % en la predicción de partidos completos mediante una

arquitectura que procesaba simultáneamente información del jugador, del partido y datos históricos.

Sin embargo, el avance más notable proviene del trabajo de Fang et al. (2024) [15], quienes aplicaron redes neuronales recurrentes (RNN) para la predicción a nivel de punto, al considerar que cada punto depende de los anteriores debido a factores como las posiciones de los jugadores en el campo. Este enfoque transformó el problema de clasificación binaria tradicional (ganar/perder) en un problema de multiclasificación que reconoce explícitamente la influencia del servicio. El modelo distingue cuatro categorías posibles: no servir y fallar, servir y fallar, no servir y ganar, y servir y ganar. Esta reformulación resultó en una mejora considerable, pasando del 71.92 % de precisión con clasificación binaria al 92.86 % con multiclasificación, demostrando que el procesamiento secuencial y la consideración del contexto del servicio mejoran significativamente la capacidad predictiva. Los autores sugieren que este enfoque puede extenderse mediante modelos jerárquicos que integren predicciones de puntos individuales para construir pronósticos a nivel de juegos, sets y partidos completos.

2.3. Análisis crítico y justificación del enfoque propuesto

El estudio conjunto del fundamento teórico y el análisis de los trabajos revisados revela importantes limitaciones en los modelos tradicionales. Por una parte, las redes neuronales feedforward presentan una arquitectura rígida que impide capturar dependencias temporales, ya que procesan cada registro de forma aislada, sin tener en cuenta la evolución a lo largo del tiempo. Esta limitación resulta especialmente restrictiva en deportes como el tenis, donde el desempeño de un jugador está estrechamente ligado a su trayectoria reciente.

Por otra parte, la mayoría de los trabajos analizados muestran un tratamiento superficial de la dimensión temporal. Los modelos convencionales tienden a ignorar la evolución secuencial del rendimiento, abordando cada partido como un evento independiente. Además, pocos estudios incorporan arquitecturas capaces de integrar tanto la información histórica como el contexto específico del encuentro.

Estas limitaciones justifican el desarrollo de un enfoque alternativo que combine redes neuronales recurrentes para el procesamiento secuencial de datos junto con variables contextuales de cada encuentro. En el tenis profesional, el rendimiento de un jugador en un partido depende en gran medida de su estado de forma reciente, por lo que es necesario emplear modelos capaces de captar patrones temporales complejos que no pueden ser detectados ni por redes feedforward ni por los enfoques tradicionales encontrados en la literatura.

Capítulo 3

Extracción, análisis y preprocesamiento de los datos

Este capítulo presenta el proceso de obtención y preparación de los datos necesarios para el desarrollo del modelo predictivo. Se detalla la implementación del sistema de extracción de datos, el análisis descriptivo de las variables obtenidas, y las transformaciones aplicadas para estructurar la información temporal y contextual requerida por las redes neuronales recurrentes.

3.1. Fuentes de datos y proceso de extracción

3.1.1. Identificación y selección de la fuente principal

Sofascore como plataforma

La primera fuente de datos elegida para este proyecto es Sofascore, una plataforma deportiva que ofrece múltiples estadísticas sobre numerosos deportes y torneos, incluyendo el tenis profesional. Esta plataforma integra información deportiva y tecnología avanzada, facilitando el acceso a datos detallados sobre jugadores y equipos [16].

Para ilustrar la interfaz web de la plataforma Sofascore, la figura 3.1 muestra la página principal de la sección de tenis en inglés (`https://www.sofascore.com/tennis`). En la captura de pantalla se aprecia la organización de la información: en la zona central aparecen los partidos correspondientes a la fecha seleccionada (15 de junio), mientras que el panel lateral izquierdo presenta algunos torneos, en la categoría "Top Competition", mostrando en primer lugar los Grand Slam.

La plataforma también permite explorar información más detallada accediendo a torneos

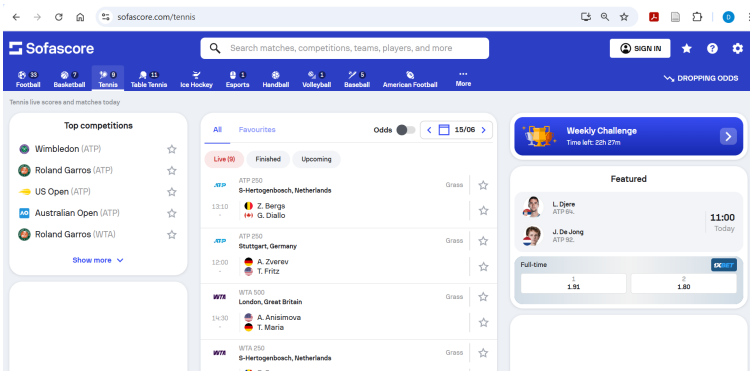

Figura 3.1: Página web principal de Sofascore en la sección de tenis el 15 de junio de 2025. Fuente: SofaScore.com

específicos, partidos concretos y perfiles de jugadores. Esta navegación se puede realizar tanto desde la página principal como utilizando el buscador integrado de la plataforma.

Para ilustrar esta funcionalidad, la figura 3.2 presenta la página web principal de Carlos Alcaraz, donde se puede acceder a información personal, estadísticas de carrera, e historial de partidos (https://www.sofascore.com/team/tennis/alcaraz-carlos/275923).

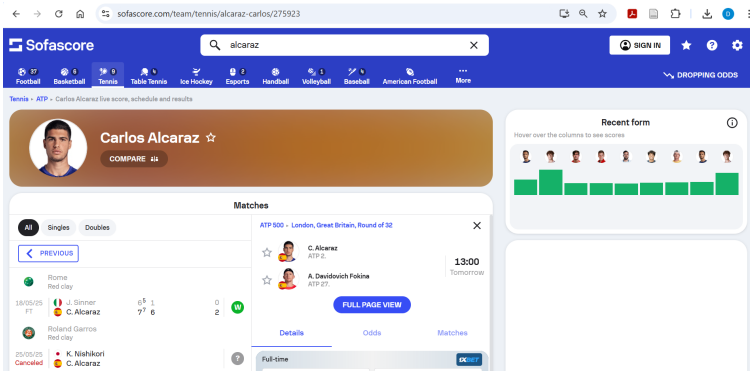

Figura 3.2: Página web del jugador Carlos Alcaraz. Fuente: SofaScore.com

Por otra parte, la figura 3.3 muestra los detalles de la final de Roland Garros del año 2025 disputada el 8 de junio. Dese esta página web es posible obtener múltiple información, como el resultado set a set, o las cuotas de las casas de apuestas, entre otros. En este encuentro, Carlos Alcaraz logró una remontada memorable al derrotar a Jannik Sinner desde un marcador de 2-0 en contra (https://www.sofascore.com/tennis/match/wsf2-wsf1/vGHbsytkc#id:13919038).

Para comprender la estructura de datos subyacente en estas páginas web, resulta esencial analizar la estructura de URLs y la forma en que Sofascore organiza sus identificadores. En la URL del perfil de Carlos Alcaraz, (https://www.sofascore.com/team/tennis/alcaraz-c

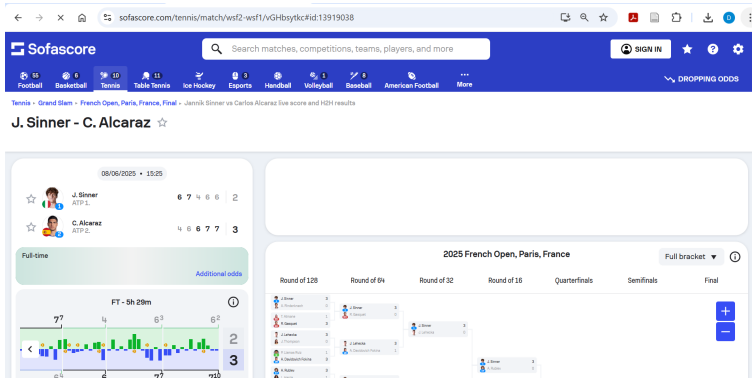

Figura 3.3: Página web correspondiente a la final de Roland Garros 2025 en Sofascore. Fuente: SofaScore.com

arlos/275923), el número "275923", constituye su identificador único dentro del sistema. De manera similar, en la URL de la final de Roland Garros (https://www.sofascore.com/tennis/match/wsf2-wsf1/vGHbsytkc#id:13919038), el valor "13919038" representa el ID específico de este partido en particular.

Esta estructura de identificadores únicos es fundamental para la posterior extracción automatizada de datos, ya que permite acceder a diferentes endpoints de una API según el tipo de información que se desee obtener.

Estructura de la API y descubrimiento de endpoints

Una API constituye una interfaz de comunicación entre diferentes sistemas software, permitiendo el intercambio de información estructurada entre ellos [17]. Concretamente, una API REST conecta varios sistemas mediante el protocolo HTTP y sus primitivas estándar, como GET y POST para obtener y crear recursos, respectivamente. En este tipo de comunicaciones, todos los recursos se manejan mediante URLs, que especifican la localización exacta del recurso en la web [18]. Los datos se intercambian en formato JSON, un estándar de representación de datos basado en objetos (pares clave-valor) y arrays (listas ordenadas de valores separadas por comas) [19]. En la figura 3.4 se muestra un ejemplo de características de Jannik Sinner en formato JSON.

Cuando un usuario accede a la página web de Sofascore, el navegador no solo se limita a cargar el contenido HTML estático visible. También se realizan numerosas peticiones automáticas a la API interna de Sofascore (https://www.sofascore.com/api/v1/) para obtener toda la información dinámica que se presenta al usuario. Esta arquitectura permite separar la estructura estática de la página (URLs y elementos visuales) del contenido dinámico (información actualizada) que se obtiene automáticamente mediante peticiones automáticas. Este patrón se repite sistemáticamente para cada sección de información de la página, desde torneos completos, hasta perfiles de jugadores individuales. La identificación de estas peticiones permite realizar una extracción automática de datos, ya que se puede

```
{
  'name': 'Jannik Sinner',
  'sport': {
    'name': 'Tennis'
  },
  'ranking': 1,
  'id': 206570
}
```

Figura 3.4: Ejemplo de datos en formato JSON

acceder directamente a la información estructurada en formato JSON, evitando la necesidad de procesar el complejo HTML.

Para identificar estas peticiones automáticas y descubrir los Endpoints específicos de la API a los que se acceden, se han utlizado las herramientas de desarrollador del navegador de Google Chrome, concretamente la pestaña "Network". Esta herramienta permite analizar todas las peticiones HTTP que se producen entre el navegador y los servidores durante la carga de una página web. Además, permite filtrar por tipo de respuesta según el recurso solicitado, por ejemplo "Doc" para documentos HTML o"CSS" para hojas de estilo. Sin embargo, para identificar los Endpoints de la API, resulta necesario aplicar el filtro "Fetch/XHR", ya que este muestra únicamente las peticiones asíncronas realizadas por JavaScript para intercambiar datos con el servidor [20]. Estas peticiones revelan los diferentes Endpoints de la API utilizados para obtener la información necesaria en formato JSON.

Para ilustrar este proceso de descubrimiento, las figuras 3.5 y 3.6 muestran un ejemplo concreto del análisis del Endpoint "unique-tournaments" mediante las herramientas de desarrollador. La figura 3.5 muestra la pestaña "Headers", que contiene los detalles técnicos de la petición HTTP. En ella, se puede observar la URL completa del Endpoint (`https://www.sofascore.com/api/v1/category/3/unique-tournaments`), información fundamental para la posterior extracción automática de los datos de los torneos. Por su parte, la figura 3.6 muestra la pestaña "Preview", que contiene la respuesta estructurada en formato JSON obtenida del servidor. La información presenta una organización jerárquica donde los torneos aparecen agrupados por mes, e incluyendo el identificador único correspondiente. Este Endpoint presenta el punto de entrada a información más específica sobre temporadas y partidos de cada torneo concreto.

El ejemplo del Endpoint de unique-tournaments, muestra una pequeña parte de una arquitectura de API más amplia y estructurada. Mediante la aplicación sistemática de la metodología de intercepción de diferentes Endpoints, se logró comprender la lógica jerárquica que se sigue en la API para organizar la información deportiva.

La organización jerárquica comienza en el nivel más alto con el Endpoint (`https://www.sofascore.com/api/v1/category/3/unique-tournaments`), que proporciona acceso a la lista completa de torneos ATP en las categorías principales (Grand Slam, Masters 1000, ATP 500, y ATP 250). La información se estructura organizando los torneos por meses del año, proporcionando de cada uno datos como el nombre o el identificador único correspondiente.

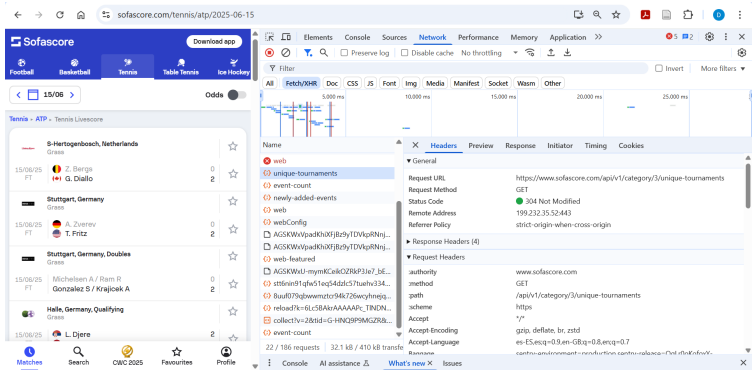

Figura 3.5: Detalles de la petición HTTP al endpoint unique-tournaments. Fuente: SofaScore.com

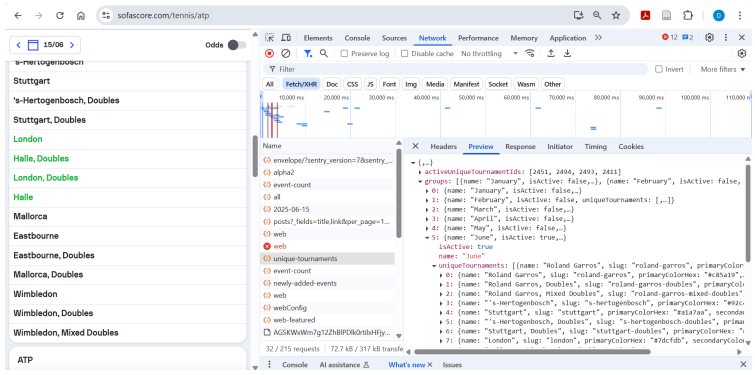

Figura 3.6: Análisis del endpoint unique-tournaments mostrando la respuesta JSON mediante herramientas de desarrollador. Fuente: SofaScore.com

En el siguiente nivel de jerarquía, se utiliza el recurso (`/api/v1/unique-tournament/{tournament_id}/seasons`) que proporciona acceso a la lista paginada de partidos disputados en una temporada específica de un torneo determinado. Este Endpoint requiere el identificador de torneo como parámetro, obtenido en el paso anterior, y devuelve la información sobre las temporadas disponibles para dicho torneo, incluyendo el identificador de temporada y el año correspondiente.

El tercer nivel se alcanza en el Endpoint `/api/v1/unique-tournament/{tournament_id}/season/{season_id}/events/last/{page}`, que proporciona acceso a una lista paginada de partidos disputados en una temporada específica de un torneo determinado. La información se organiza cronológicamente desde los eventos más antiguos (página 0, correspondiente a las primeras rondas) hasta los eventos más recientes (páginas más altas, que incluye las rondas finales). Este Endpoint resulta fundamental para la extracción automática de datos, ya que obtiene los identificadores de todos los partidos disputados en un contexto temporal específico.

Para obtener los detalles de cada partido individual, se utiliza el punto de acceso (`/api/v1/event/{match_id}`), que constituye una de las fuentes más ricas del sistema. Este Endpoint devuelve datos completos del encuentro, desde información contextual del partido (superficie del juego, ronda o categoría) hasta detalles como el resultado por set y los identificadores de los jugadores participantes. Esta amplia gama de información convierte a este punto de acceso en el componente principal del sistema.

Complementariamente, se accede al Endpoint de probabilidades `https://www.sofascore.com/api/v1/event/{match_id}/odds/1/featured` para obtener las cuotas de apuesta de cada jugador previas al encuentro. Este recurso proporciona información sobre las expectativas del mercado, incluyendo la cuota inicial para ambos participantes.

A nivel de jugador, el Endpoint `/api/v1/team/{player_id}` proporciona datos personales y estadísticas de un jugador en específico, como país de origen, fecha de nacimiento y ranking actual, correspondiendo este al valor más reciente del jugador. El parámetro "player_id" se obtiene a partir de la información devuelta por el Endpoint de los partidos individuales. Por otra parte, `/api/v1/team/{player_id}/events/last/{page}` permite acceder al historial paginado de partidos disputados por un jugador en concreto. Este punto de acceso devuelve información estructurada de manera similar al de partidos individuales, facilitando la obtención del registro competitivo de cada tenista.

3.1.2. Implementación del proceso automatizado de extracción

Una vez identificada la estructura jerárquica de los Endpoints de la API de Sofascore, se procedió a implementar un sistema automatizado de extracción de datos que replicase el proceso descubierto mediante las herramientas de desarrollador. Esta implementación se realizó en Python, lenguaje de programación que incluye librerías especializadas en web scraping y manipulación de datos.

La librería principal utilizada para las peticiones HTTP fue "requests". Este módulo permite realizar peticiones de tipo "GET" de manera sencilla, ya que se encarga automáticamente de establecer la conexión, enviar la petición HTTP correspondiente, y devolver un objeto respuesta que incluye tanto el contenido en formato JSON, como metadatos relevantes de la petición, como el estado o el tiempo de respuesta. Por ejemplo, una petición típica se ejecutaría simplemente mediante la función `requests.get(url)`. Complementariamente se utilizó Pandas para la estructuración y manipulación de los datos extraídos, permitiendo organizar la información en DataFrames que facilitan el análisis y procesamiento de éstos.

El enfoque metodológico sigue estrictamente la jerarquía de Endpoints identificada, implementando una función específica para cada nivel dentro de la arquitectura de datos. De esta manera, cada módulo corresponde a un paso específico del proceso de navegación a través de la API.

La función `scraping_tournaments()` constituye el punto de entrada al sistema, accediendo al Endpoint `https://www.sofascore.com/api/v1/category/3/unique-tournaments` para obtener la lista completa de torneos ATP disponibles en la plataforma. Esta función selecciona únicamente los torneos de las categorías principales del circuito ATP para centrar

el análisis en los eventos de mayor relevancia competitiva y garantizar un volumen de datos manejable, identificándolos mediante el campo `tennisPoints`, que debe contener los valores 500, 1000 o 2000, correspondientes a las categorías de ATP 500, Masters 1000 y Grand Slam. Adicionalmente, se incluye el torneo ATP Finals, que por sus características especiales no sigue el patrón estándar de puntuación. Además, se excluyen aquellos torneos cuyo nombre contenga las palabras "Double" o ´´Doubles", asegurando que el sistema procese únicamente torneos individuales.

A continuación, la función `scraping_seasons()` opera sobre la lista de torneos obtenida anteriormente, utilizando el Endpoint `/unique-tournament/{id}/seasons` para cada torneo en específico con el objetivo de obtener las temporadas específicas. Esta función requiere como parámetro el año de interés, permitiendo filtrar por temporada. Adicionalmente, requiere ajustes manuales, ya que el calendario ATP experimenta variaciones entre diferentes temporadas como modificaciones en la categoría o cambios de sede.

La función `scraping_id_matches()` procesa los pares torneo-temporada obtenidos accediendo al Endpoint `/api/v1/unique-tournament/{tournament_id}/season/{season_id}/events/last/{page}` para extraer los identificadores de todos los partidos disputados. Esta función implementa un sistema de paginación que recorre las páginas numeradas asegurando la obtención de todos los eventos. El resultado es una lista de identificadores únicos de partido que servirá como entrada para la extracción de información detallada de cada encuentro.

El método `scraping_partido()` representa el componente más importante del sistema, ya que accede al Endpoint `/event/{match_id}` para obtener información de cada partido individual. Esta función extrae una amplia gama de datos, que incluyen información contextual del encuentro (superficie, ronda, categoría), detalles específicos del partido (resultado por set, resultado final, estado), datos de ambos jugadores participantes y as probabilidades de victoria asignadas por las casas de apuestas a cada jugador. Esta información constituye una variable predictiva adicional que refleja las posibilidades de éxito de cada deportista.

Para cada jugador identificado, se realiza una llamada a la función `get_player()` con su identificador correspondiente. Esta función gestiona dos tipos de información: estática y dinámica. La información estática (características físicas, fecha de nacimiento, país de origen) se obtiene mediante el Endpoint `/team/{player_id}`, extrayendo datos personales que permanecen constantes a lo largo de su carrera profesional. Pese a que variables como la altura y el peso pueden presentar ligeras variaciones según el momento en el que se midan, se consideran constantes para el análisis predictivo.

Por su parte, la información dinámica se refiere al ranking del jugador en el momento en el que se disputa el partido. Este dato varía temporalmente, requiriendo un tratamiento diferenciado para reflejar la clasificación real en cada fecha concreta. En este momento surge una importante limitación, ya que el Endpoint de Sofascore devuelve el ranking actual del jugador y su mejor ranking histórico a fecha de consulta, sin proporcionar la clasificación que tenía en el momento del partido.

La limitación identificada en Sofascore respecto a la información temporal del ranking supone un problema significativo, ya que éste constituye uno de los indicadores más importantes a nivel competitivo para predecir la probabilidad de victoria. Esta limitación motivó

la búsqueda de una fuente de datos complementaria que proporcionase el historial de clasificaciones por fecha.

3.1.3. Fuente complementaria: Matchstat

Matchstat constituye una plataforma de información deportiva que, aunque comparte similitudes con Sofascore en cuanto a los datos del circuito ATP, se diferencia en el mantenimiento de registros históricos de clasificaciones. Esta plataforma proporciona los rankings de los 900 primeros jugadores del circuito ATP masculino desde 1973, permitiendo consultar la posición de cualquier tenista en una fecha específica del pasado. La figura 3.7 ilustra la interfaz de acceso a esta información temporal (https://matchstat.com/tennis/atp-wta-rankings/atp), mostrando la clasificación correspondiente al 5 de mayo de 2025.
Al igual que Sofascore, Matchstat también se sustenta en una API interna que estructura el acceso a la información mediante Endpoints específicos.

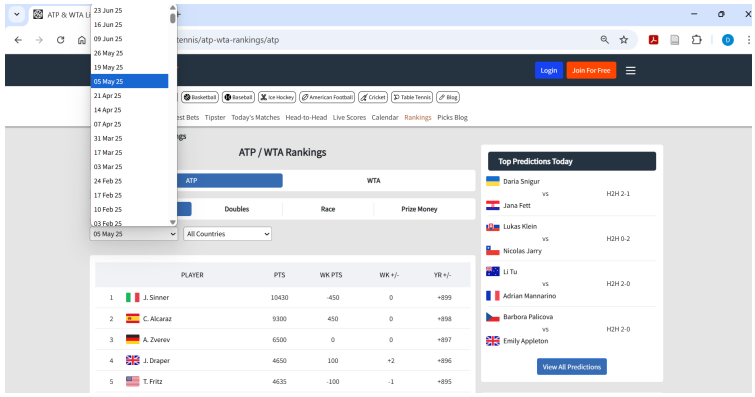

Figura 3.7: Interfaz de rankings históricos por fechas en Matchstat, mostrando la selección temporal en la parte izquierda y la clasificación. Fuente: Matchstat.com

Implementación del proceso de extracción de los rankings

La principal diferencia técnica en el proceso de extracción respecto a Sofascore radica en las protecciones anti-bot implementadas por Matchstat mediante Cloudflare. Cloudflare constituye un servicio de gestión de bots que protege los sitios web contra el tráfico automatizado no deseado, implementando verificaciones que distinguen entre usuarios reales y programas automatizados [21]. Esta protección impide el uso de la librería "requests" , ya que las peticiones automáticas son detectadas y bloqueadas por el sistema. Es por ello que las peticiones se realizan mediante la librería "cloudscraper".

El proceso de extracción de rankings históricos se desarrolló en un script independiente, separando esta funcionalidad del sistema de extracción de Sofascore. En este archivo, la

función `obtener_ranking()` coordina el proceso de obtención y estructuración de la información.

El proceso comienza accediendo al Endpoint https://matchstat.com/tennis/api2/ranking/atp/filters?includeAll=true para obtener la lista completa de fechas en las que se han producido actualizaciones del ranking ATP, normalmente con frecuencia semanal. De esta lista se seleccionan únicamente las fechas posteriores al 12 de enero de 2009.

Para cada fecha seleccionada, se accede al Endpoint https://matchstat.com/tennis/api2/ranking/atp/?date={date}&page={page} mediante un sistema de paginación similar al implementado en Sofascore. Este proceso extrae el nombre y la posición en el ranking de cada jugador en la fecha especificada. Si el jugador ya aparece en el DataFrame por fechas anteriores, simplemente se actualiza la columna correspondiente. En caso contrario, se crea una nueva fila para el jugador. Cuando se identifica un jugador por primera vez, se consulta el Endpoint https://matchstat.com/tennis/api2/profile/{player_name} para obtener información personal complementaria, fecha de nacimiento (convertida a formato timestamp) y país de origen.

Una vez completada la extracción, se aplica un proceso de normalización que incluye la conversión de códigos de país al formato alpha-3 estándar [22], ya que Matchstat utiliza nomenclaturas propias. Los nombres de jugadores se estandarizan mediante la eliminación de caracteres especiales y la aplicación del formato nombre-apellidos. Finalmente se corrigen manualmente errores específicos detectados tras múltiples ejecuciones, como nacionalidades incorrectas o fechas de nacimiento erróneas.

El resultado final de este proceso es un DataFrame estructurado donde las filas representan todos los jugadores del top 900 del circuito ATP desde el enero de 2009 hasta la fecha actual. Las columnas incluyen nombre completo, fecha de nacimiento y nacionalidad, seguidas de columnas temporales identificadas por timestamps. Cada celda contiene la posición de ranking correspondiente a esa fecha, utilizando el valor -1 para indicar períodos en los que el jugador no figuraba entre los 900 primeros clasificados. La tabla 3.1 muestra un ejemplo de esta estructura con los tres primeros jugadores del ranking y cuatro fechas consecutivas en 2025.

Player	**birthDate**	**country**	**1743976800**	**1743372000**	**1742166000**	**1740956400**
sinner-jannik	997912800	ITA	1	1	1	1
zverev-alexander	861487200	DEU	2	2	2	2
alcaraz-carlos	1052085600	ESP	3	3	3	3

Tabla 3.1: Ejemplo del DataFrame de ranking histórico mostrando cuatro fechas consecutivas desde el 3 de marzo al 7 de abril de 2025, con las fechas de nacimiento de los jugadores

3.1.4. Integración y construcción final de los DataSets

Integración de las fuentes de datos

Una vez obtenidos los datos de partidos desde Sofascore y los rankings históricos desde Matchstat, surge la necesidad de integrar ambas fuentes de información. Este proceso requiere establecer correspondencias precisas entre los nombres de jugadores, que presentan diferencias según la plataforma.

El primer paso consiste en aplicar el mismo proceso de normalización de nombres utilizado en Matchstat a los datos obtenidos de Sofascore. Sin embargo, las coincidencias exactas son improbables debido a las diferencias en la información de cada fuente. Por ejemplo, el jugador que aparece en Sofascore como "m-topo"se registra en Matchstat como "Marko Topo", que tras la normalización se convierte en "marko-topo".

Para resolver estas diferencias, inicialmente se implementó un sistema basado en la librería "rapidfuzz", que utiliza algoritmos de matching para encontrar correspondencias entre cadenas de texto mediante puntuaciones de similitud [23]. Sin embargo, aplicar este método sobre todo el conjunto de datos generaba numerosos falsos positivos, confundiendo jugadores con apellidos comunes como "Jorge Martínezz "Pedro Martínez".

Tras múltiples pruebas, se implementó una solución basada en un sistema de filtrado que combina el matching con criterios adicionales de verificación. Para cada jugador de Matchstat que necesite correspondencia, se filtran los jugadores de Sofascore que cumplan dos condiciones: fecha de nacimiento dentro de un rango de 3 días de diferencia y nacionalidad idéntica.

Esta estrategia de filtrado reduce las opciones disponibles, permitiendo disminuir el umbral de puntuación requerido para establecer correspondencias válidas. Este enfoque aumenta la sensibilidad del sistema sin comprometer la precisión, ya que las coincidencias casuales en fecha de nacimiento y nacionalidad son prácticamente imposibles. Los casos en los que no se establece correspondencia indican que el jugador no existe en la base de datos de Matchstat, lo cual significa que nunca ha alcanzado el top 900 del ranking ATP.

Con este enfoque, se consigue integrar ambas fuentes de datos, obteniendo para cada jugador tanto su información personal básica como su evolución en el ranking a lo largo del tiempo. Esta combinación proporciona el conjunto completo de variables necesarias para analizar en profundidad las características de cada partido disputado.

Construcción de los datasets finales

El sistema está diseñado para generar dos conjuntos de datos que se complementan en el proceso de predicción. El DataSet "actual" contiene los partidos cuyo resultado se desea predecir, incluyendo únicamente información disponible antes del encuentro: características de ambos jugadores, superficie, torneo, probabilidades otorgadas por las casas de apuestas y variables contextuales conocidas previamente. Por su parte, el DataSet "previos" almacena

el historial competitivo de cada jugador que participa en los encuentros del DataSet "Actual". Este conjunto incluye tanto las condiciones iniciales de cada partido histórico como la información de resultados: marcadores por set, resultado global, y variables adicionales como el tiempo transcurrido desde el partido anterior.

Para la construcción de estos DataSets, el sistema procesa secuencialmente cada identificador de partido de la lista obtenida de Sofascore, construyendo ambos conjuntos de datos de forma simultánea. El proceso comienza extrayendo la información previa de cada encuentro, que luego se evalúa mediante varios filtros de calidad.

Estos criterios incluyen la ausencia de valores nulos en variables esenciales, la confirmación de que se trata de un partido individual (excluyendo dobles), la exclusión de encuentros de Copa Davis (que no forman parte del circuito ATP), y la comprobación de que el partido esté completamente finalizado (indicado por el estado 100). Adicionalmente, se requiere que ambos jugadores tengan alguna posición en el ranking ATP, es decir, que su clasificación sea diferente de -1 en el DataSet de ranking correspondiente a la fecha del partido.

Durante este proceso de filtrado, algunos partidos presentan información incompleta en las variables físicas de los jugadores. En casos donde falta únicamente la altura o peso de un jugador, se implementa un sistema de imputación mediante regresión lineal que aprovecha la correlación natural entre ambas variables para estimar el valor faltante. La imputación mediante regresión lineal se implementa para aprovechar la fuerte correlación existente entre altura y peso ($r = 0.748$), permitiendo estimar el valor faltante basándose en la relación natural entre ambas variables. Este enfoque resulta más preciso que métodos simples como la imputación por media o mediana, ya que mantiene la dependencia entre las características físicas y evita introducir valores que no reflejen las proporciones reales de los jugadores. En el caso de la lateralidad, se asigna automáticamente "diestro", dada su mayoría estadística. La imputación de valores faltantes permite minimizar el sesgo introducido por la eliminación de casos incompletos.

Si el partido supera los criterios mencionados, se procede a construir el historial de cada jugador accediendo el Endpoint `https://www.sofascore.com/api/v1/team/{id_player}/events/last/{page}`. Para los partidos históricos se aplican criterios de filtrado más permisivos, permitiendo que alguno de los jugadores no esté clasificado en el ranking, pero exigiendo información completa de resultados.

La estructura de datos establece una correspondencia específica entre ambos conjuntos, imponiendo como requisito obtener exactamente 50 partidos previos válidos para cada jugador del encuentro actual. Si no se alcanza este número para cualquiera de los participantes, se descarta tanto el partido actual como todo su historial asociado.

La organización temporal sigue el orden cronológico natural, desde los partidos más antiguos hacia los más recientes. Para cada encuentro del DataSet "actual" existen exactamente 100 partidos asociados en "previos", donde los primeros 50 corresponden al historial del jugador local y los siguientes 50 al jugador visitante. Esta estructura se mantiene sistemáticamente a lo largo de todo el DataSet, por lo que si "actual" tiene N partidos para predecir el resultado,"previos" tendrá exactamente $N \times 100$ partidos históricos, organizados en bloques de 100 filas.

La figura 3.8 ilustra esta organización mediante un ejemplo práctico con partidos de Roland Garros 2024, mostrando cómo los encuentros fluyen temporalmente entre ambos datasets. Esta estructuración facilita el trabajo posterior con redes neuronales recurrentes, garantizando que cada predicción cuente con el mismo número de partidos históricos y permitiendo que el modelo procese secuencias de longitud fija.

3.2. Análisis descriptivo de los datos

3.2.1. Estructura y características generales de los Datasets

Una vez desarrollado el sistema de extracción, se ejecutó el programa para recopilar información de los torneos principales del circuito ATP, abarcando las categorías ATP 500, Masters 1000, Grand Slam y ATP Finals durante las temporadas 2021, 2022, 2023 y 2024. El proceso inicial recopiló un total de 9.878 partidos en el DataSet "Actual", aunque tras la aplicación de los filtros de calidad se redujo a 9.259 partidos válidos, representando una exclusión del 6,3 %.

La mayoría de las eliminaciones corresponden a partidos que no alcanzaron el estado de finalización completa (suspendidos, abandonos o pase directo), mientras que únicamente 188 partidos se descartaron por no cumplir el requisito de contar con al menos 50 partidos previos válidos para ambos jugadores. De esta manera, el DataSet "actual" contiene 9.259 partidos con 27 variables, y el DataSet "previos" contiene 925.900 registros históricos distribuidos en 41 variables.

Los conjuntos de datos extraídos presentan una estructura que combina variables comunes con información específica según el contexto, diferenciando entre el dataset de predicción (actual) y el histórico (previos). La mayoría de las variables se organizan en pares correspondientes a los jugadores local (Home) y visitante (Away), cuyas características deben analizarse de forma simétrica.

Las variables comunes a ambos conjuntos de datos incluyen, en primer lugar, las variables identificativas (`idTournament`, `idSeason`, `idEvent`) que permiten la identificación única de cada encuentro, complementadas con el nombre del torneo (`tournamentName`) y la ronda disputada, esta última codificada numéricamente.

Las variables contextuales incluyen información sobre las condiciones del encuentro. El tipo de superficie (`groundType`) puede tomar cuatro valores posibles: Hardcourt Outdoor, Hardcourt Indoor, Red Clay y Grass, correspondientes a los cuatro tipos de superficie posibles para el tenis profesional. El número de sets (`periodCount`) indica la modalidad del encuentro, siendo normalmente al mejor de 3 sets, aunque en los Grand Slam y algunas excepciones específicas se disputa al mejor de 5. Por otra parte, el timestamp de inicio (`startTimestamp`) registra la fecha y hora exacta en que se disputó el encuentro. Finalmente, la variable de estado (`status`) confirma que el partido finalizó correctamente, ya que únicamente se incluyen encuentros con valor 100, indicando la finalización completa.

La información de cada jugador se estructura simétricamente diferenciando entre el

Estructura completa de datos

Dataset "Actual" (partidos a predecir)

Torneo	Ronda	JugadorHome	JugadorAway
Roland Garros 2024	Semifinal	Alcaraz	Sinner
Roland Garros 2024	Final	Alcaraz	Zverev

1:100
50 por jugador

Bloque 1 de "Previos" - Semifinal Alcaraz vs Sinner (filas 1-100)

Historial de Alcaraz (filas 1-50):

Torneo	Ronda	Home	Away	Resultado
Viena 2023	Cuartos	Alcaraz	Rublev	1-2
⋮ *(47 partidos mas de Alcaraz)* ⋮				
Roland Garros 2024	Octavos	Alcaraz	Auger-A.	3-0
Roland Garros 2024	Cuartos	Alcaraz	Tsitsipás	3-0

Historial de Sinner (filas 51-100):

Torneo	Ronda	Home	Away	Resultado
Cincinnati 2023	Cuartos	Purcell	Sinner	1-2
⋮ *(47 partidos mas de Sinner)* ⋮				
Roland Garros 2024	Octavos	Moute	Sinner	1-3
Roland Garros 2024	Cuartos	Dimitrov	Sinner	0-3

Bloque 2 de "Previos" - Final Alcaraz vs Zverev (filas 101-200)

Historial de Alcaraz (filas 101-150):

Torneo	Ronda	Home	Away	Resultado
Viena 2023	Semifinal	Alcaraz	Medvedev	0-2
⋮ *(47 partidos mas de Alcaraz)* ⋮				
Roland Garros 2024	Cuartos	Alcaraz	Tsitsipás	3-0
Roland Garros 2024	Semifinal	Alcaraz	Sinner	3-2

Historial de Zvrev (filas 151-200):

Torneo	Ronda	Home	Away	Resultado
Roma 2023	Octavos	Medvedev	Zvrev	2-0
⋮ *(47 partidos mas de Zverev)* ⋮				
Roland Garros 2024	Cuartos	de Minaur	Zvrev	0-3
Roland Garros 2024	Semifinal	Ruud	Zverev	1-3

Figura 3.8: Estructura organizativa de los DataSets "actual" y "previos" mostrando la relación 1:100 entre partidos a predecir y su historial correspondiente. Los datos se organizan cronológicamente desde partidos más antiguos (primera fila) hasta el más reciente (última fila). Los partidos del mismo color representan el mismo encuentro: los rojos aparecen en "actual" cuando se predicen y en "previos" como historial para partidos posteriores; los azules se repiten entre diferentes bloques al ser partidos comunes en el historial de Carlos Alcaraz.

competidor local (Home) y visitante (Away). Para cada participante se registra su identificador único en Sofascore (`idHome/Away`), la fecha de nacimiento en formato timestamp (`birthDateHome/Away`), rasgos físicos como altura y peso (`HeightHome/Away`, `WeightHome/-Away`, y lateralidad (`RightHandedHome/Away`). Las variables de clasificación incluyen el ranking en el momento del partido (`ActualRankingHome/Away`) y el mejor ranking histórico alcanzado hasta esa fecha (`BestRankingHome/Away`).

Finalmente, ambos DataSets comparten la variable objetivo `winnerCode`, que toma el valor 0 cuando gana el jugador local y 1 cuando gana el visitante. En el DataSet "actual", esta variable representa el target (objetivo) que permitirá evaluar la precisión de las predicciones del modelo. Adicionalmente, este conjunto de datos incorpora las probabilidades de victoria asignadas por las casas de apuestas antes del partido (`ProbabilityHome/Away`), que constituyen una variable predictiva complementaria al ganador del partido. Estas probabilidades reflejan la expectativa del mercado de apuestas sobre qué jugador tenía más opciones de ganar antes de que comenzara el partido.

El DataSet histórico incluye información temporal adicional como `lastMatchTimestamp`, que registra el momento en que se disputó el partido anterior del jugador correspondiente, permitiendo calcular el tiempo transcurrido entre encuentros consecutivos. También incorpora el identificador del siguiente partido válido en el historial (`idNext`), permitiendo establecer la secuencia cronológica de partidos. Finalmente, las variables de rendimiento constituyen la mayor diferencia respecto al DataSet "actual", proporcionando información detallada sobre el desarrollo de cada encuentro: el marcador final (`homeScore/awayScore`), el resultado específico de cada set disputado (`set1performanceHome/Away` hasta `set5performanceHome/Away`), y el total de juegos ganados por cada participante (`totalGamesHome/Away`).

3.2.2. Variables contextuales

El conjunto de datos abarca el período 2021-2024, mostrando una distribución temporal con crecimiento progresivo en el registro de partidos. El incremento observado en años más recientes se debe a una mayor disponibilidad de historiales completos de partidos, lo que facilita el cumplimiento de los criterios mínimos de partidos previos establecidos. Esta distribución temporal equilibrada garantiza la representatividad del DataSet, minimizando posibles sesgos temporales. La tabla 3.2 detalla la distribución anual de los partidos analizados.

Año	**Partidos**
2021	2071
2022	2226
2023	2509
2024	2453
total	9259

Tabla 3.2: Distribución temporal de partidos en el DataSet por año de competición (2021-2024)

La distribución de torneos revela una estructura jerárquica propia del circuito profesional. Los cuatro Grand Slam (Roland Garros, Wimbledon, Australian Open y US Open) dominan el DataSet con aproximadamente 900 partidos válidos cada uno, lo que refleja tanto su relevancia en el circuito ATP como su mayor duración y formato extendido. En contraste, los torneos de categoría inferior presentan frecuencias menores, conformando una distribución representativa de la estructura del tenis profesional, donde una minoría de eventos concentra a los mejores tenistas. La figura 3.9 ilustra la distribución de partidos por torneo.

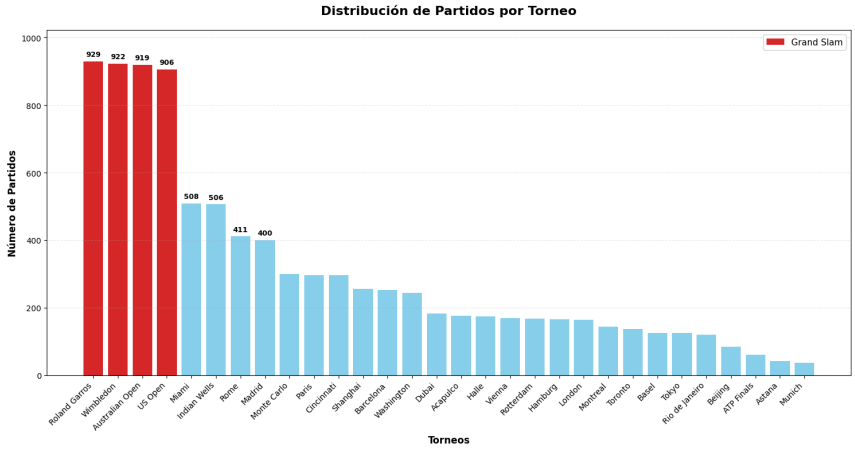

Figura 3.9: Distribución de partidos por torneo en el DataSet ordenados por orden de frecuencia descendente.

Por último, el análisis de superficies de juego revela una distribución que refleja fielmente la realidad del circuito profesional. Como se observa en la figura 3.10, la pista dura exterior domina con un 48.8 % de los encuentros, seguida por la tierra batida (28.2 %), hierba (13.6 %)y pista dura cubierta (9.3 %). Esta distribución es especialmente relevante para el modelo predictivo, ya que cada superficie presenta características técnicas diferenciadas.

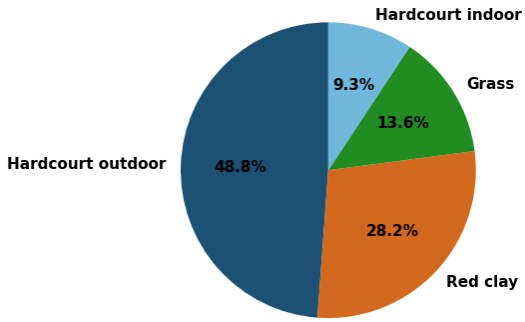

Figura 3.10: Distribución de partidos por tipo de superficie de juego.

3.2.3. Variables de rendimiento de jugadores

Para analizar las variables de ranking, se examinan conjuntamente `ActualRankingHome-/Away` y `BestRankingHome/Away`, ya que representan la misma variable aplicada a diferentes jugadores del partido. Es importante mencionar que se invierte la escala tradicional de ATP, de modo que el valor 1 representa al jugador con peor clasificación y el 900 al mejor. Esta codificación facilita la interpretación, ya que valores más altos indican mejor posicionamiento en el ranking.

Los resultados descriptivos de la tabla 3.3 muestran características singulares: Ambas variables presentan asimetría fuertemente negativa (actualRanking: -3.387, bestRanking: -3.12) y curtosis muy elevada (actual: 18.19, mejor: 17.3). La asimetría negativa supone que la mayoría de datos se agrupan en los valores altos (buenas posiciones), con pocos casos dispersos hacia valores bajos. La curtosis elevada indica que los datos están muy concentrados en una zona específica, en lugar de distribuirse de forma equilibrada .

	Actual Ranking	**Best Ranking**
Media	798.86	836.03
Desviación típica	112.7	76.37
Mediana	828	861
Asimetría	-3.387	-3.12
Curtosis	18.19	17.3

Tabla 3.3: Estadísticos descriptivos de las variables de ranking. Se observa concentración hacia los jugadores con posiciones más altas.

Esta concentración extrema se visualiza en la figura 3.11, donde ambos histogramas muestran una alta acumulación de jugadores en los rangos superiores (800-900). La magnitud de esta desproporción se evidencia en la figura 3.12, que muestra que aproximadamente 11.000 registros corresponden al top 100 de jugadores (rangos 800-900), mientras que el resto de niveles tienen representaciones drásticamente menores.

Esta situación es natural en el conjunto de datos, ya que se incluyen los torneos de las categorías más altas del circuito ATP donde solo los mejores jugadores participan. Además, los jugadores con mejor ranking avanzan más rondas y generan más registros en el DataSet. Por ejemplo, si Djokovic llega a semifinales aporta 6 partidos al DataSet, mientras que un jugador con ranking 100 (top 800) eliminado en la primera ronda solo aporta 1 partido. Estos dos hechos explica la alta representación de los mejores tenistas en el conjunto de datos.

Debido a que el objetivo del modelo es predecir el resultado de un partido entre dos jugadores, es interesante explorar cómo se comportan las diferencias de rankings entre los dos adversarios. Por ejemplo, conocer que existe una diferencia de 50 posiciones entre dos adversarios (850 vs 800) podría ser más informativo para el modelo que conocer únicamente sus posiciones individuales.

Sin embargo, el análisis de las diferencias absolutas entre rankings revela una problemática que implica la necesidad de transformaciones adicionales: las diferencias absolutas reflejan

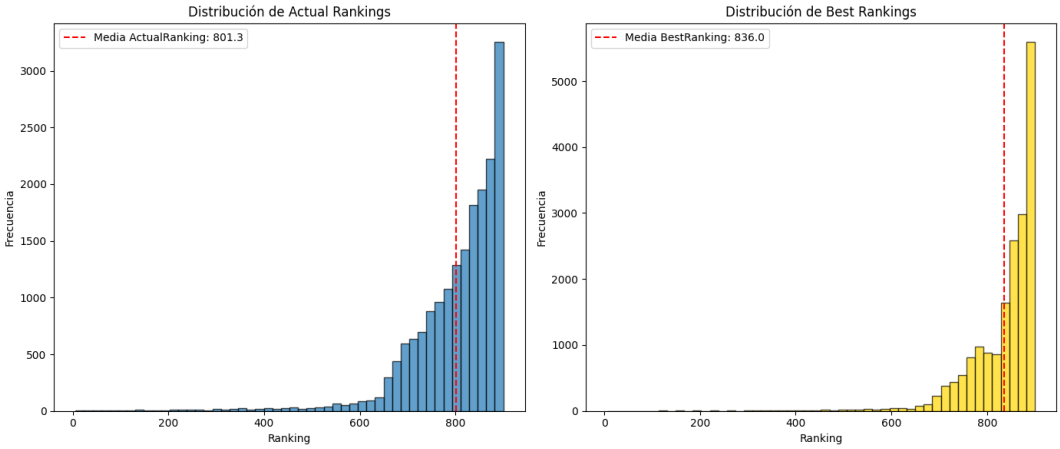

Figura 3.11: Histograma de ranking actual y mejor ranking histórico. Se observa concentración extrema hacia valores altos, con asimetría negativa pronunciada.

efectos diferentes según el nivel de los jugadores involucrados. Para analizar esta problemática, se clasifican los partidos según el promedio de los rankings de ambos jugadores: partidos de nivel Medio (promedio entre 0 y 700), Alto (700-850) y Élite (850-900).

La figura 3.13 muestra cómo se distribuyen las diferencias absolutas de ranking según el nivel del partido, observándose patrones claramente diferenciados entre los tres niveles. El nivel Élite (verde) presenta diferencias concentradas en un rango muy estrecho alrededor del cero, lo cual es lógico ya que ambos jugadores pertenecen al top 50 mundial y las diferencias de ranking entre ellos son menores. Por contra, el nivel Alto (naranja) muestra una distribución más amplia, con diferencias que pueden alcanzar las 300 posiciones. Por último, el nivel Medio (azul) presenta escasa representación en el DataSet, lo que confirma una vez más que los datos están muy concentrado en los jugadores con mejor clasificación.

La figura 3.14 revela que las mismas diferencias absolutas generan tasas de victoria diferentes según el nivel competitivo. Examinando la línea roja vertical, que marca una diferencia de +50 posiciones (el jugador home supera en 50 puestos al away), se observa que en el nivel Elite supone tasas de victoria cercanas al 75-80 %, mientras que en los niveles Alto y Medio, la misma diferencia produce tasas del 55-60 %.

Además, las curvas muestran comportamientos claramente diferenciados: el nivel Elite presenta pendientes muy pronunciadas donde pequeñas diferencias de ranking generan grandes cambios en la probabilidad de victoria, mientras que los otros niveles se observan transiciones más suaves y graduales. Estas diferencias demuestran que el contexto competitivo modifica el significado de las diferencias en el ranking. En otras palabras, una diferencia de 50 posiciones entre dos jugadores top 100 tiene diferentes implicaciones que la misma diferencia entre dos jugadores con puestos inferiores, aunque numéricamente sea idéntica.

En línea con el enfoque adoptado para las variables de ranking, el análisis de las características físicas se centra en las diferencias entre adversarios en altura y peso. La tabla 3.4

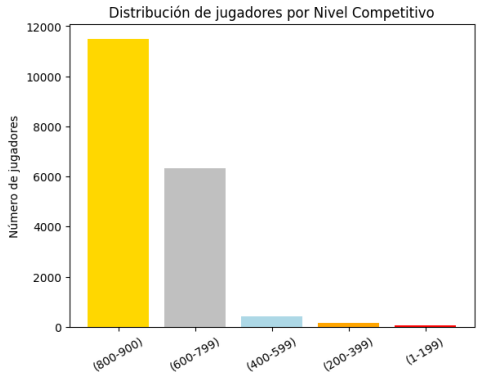

Figura 3.12: Distribución de jugadores por nivel competitivo. Se observa la concentración extrema en el top 100 (800-900)

presenta los estadísticos descriptivos de las diferencias físicas, revelando medidas favorables: Ambas variables exhiben medias prácticamente nulas, asimetría mínima y curtosis baja, lo que indica distribuciones simétricas y centradas en cero. Esta característica confirma la ausencia de sesgos que favorezcan a jugadores home o away en términos de ventajas físicas.

	Diferencia Altura	**Diferencia Peso**
Media	-0.000	-0.083
Desviación típica	0.095	9.027
Asimetría	0.013	0.015
Curtosis	0.168	0.534

Tabla 3.4: Estadísticos descriptivos de las diferencias físicas entre jugadores. Se observa simetría y centrado en cero

Estas medidas descriptivas establecen una base para las decisiones de preprocesamiento que se implementarán posteriormente

3.2.4. Variables objetivo y probabilidades

La variable ProbabilityHome representa las probabilidades que asigna el mercado de apuestas al jugador home, constituyendo una fuente relevante para el modelo predictivo. El análisis descriptivo revela que la probabilidad promedio es del 54.1 %, evidenciando una ligera ventaja del jugador home. La suma promedio de ambas probabilidades (home + away) alcanza 1.058, lo que implica un "vigorish" del 5.8 % para las casas de apuestas, margen que se encuentra dentro del rango estándar [24].

Un aspecto relevante es la alta eficiencia predictiva del mercado, con un acierto del 68.8 %, donde las probabilidades asignadas por las casas coinciden con los resultados reales. Esta

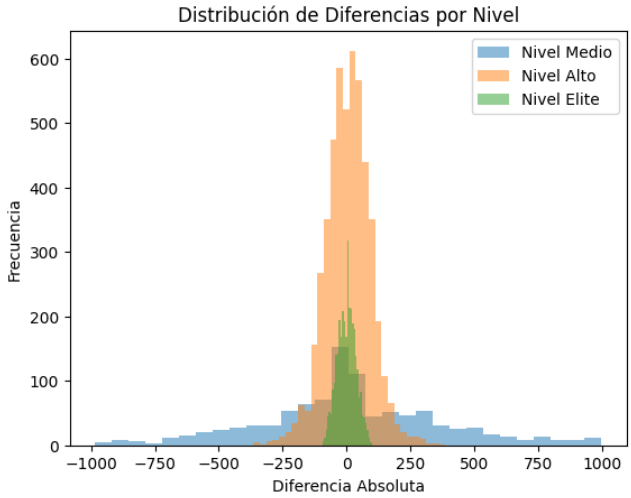

Figura 3.13: Distribución de diferencias absolutas de ranking según el nivel competitivo del partido. Se observan patrones diferenciados: concentración estrecha en Elite, mayor dispersión en Alto y escasa representación en Medio

precisión convierte a las probabilidades de la casa de apuestas en una variable especialmente valiosa para el desarrollo del modelo predictivo.

El análisis de la variable objetivo (`winnerCode`) revela que el DataSet está balanceado, con victoria del jugador home en el 51.0 % de los casos frente al 49.0 % del jugador away. Este equilibrio elimina preocupaciones sobre sesgos en los datos.

Al examinar la distribución de victorias por tipo de superficie, la figura 3.15 muestra que la ventaja del jugador home presenta ligeras variaciones según el terreno de juego. La hierba exhibe la mayor ventaja local (55.2 %), seguida por pista dura cubierta, dura exterior, y tierra batida, que muestran distribuciones más equilibradas. Aunque las diferencias son moderadas, sugieren que el tipo de superficie puede interactuar con otros factores contextuales del partido, lo que podría ser relevante para el modelo predictivo.

3.2.5. Variables históricas

El conjunto de datos históricos proporciona el contexto temporal necesario para el modelo predictivo, con 925.900 registros que documentan el historial previo al partido.

El análisis de duración de partidos revela que el 56.3 % de encuentros se resuelven en 2 sets (520,958 registros), mientras que el 37.5 % requieren 3 sets (346,939 registros). Los encuentros más largos son menos frecuentes, el 3.8 % llegan a 4 sets y apenas el 2.4 % se extienden hasta 5 sets.

Adicionalmente, el DataSet histórico incluye información temporal que permite calcular

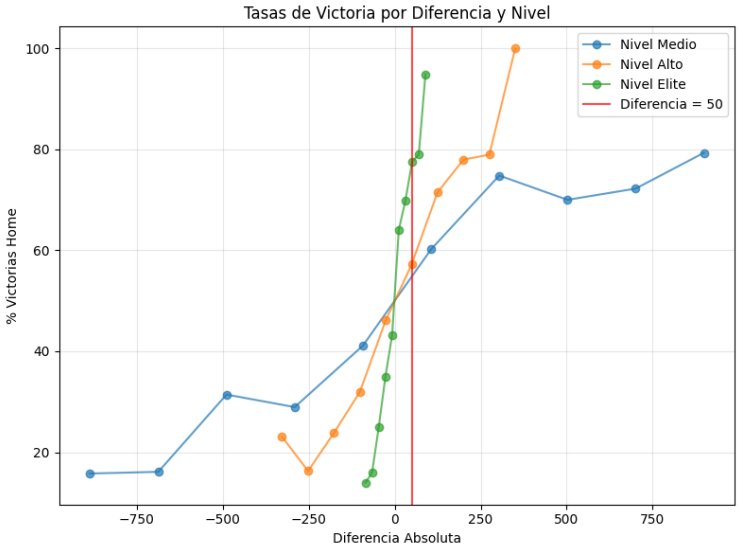

Figura 3.14: Tasas de victoria del jugador home según la diferencia absoluta de ranking y nivel competitivo. La línea roja (diferencia = +50) muestra que la misma diferencia absoluta genera tasas de victoria diferentes según el nivel competitivo

el tiempo transcurrido entre partidos consecutivos de cada jugador mediante la diferencia entre timestamps. Esta variable revela patrones interesantes, como que los jugadores suelen competir con intervalos cortos (mediana de 2 días), aunque la media de 5.6 días refleja la influencia de períodos más largos de descanso entre torneos. Los períodos de inactividad pueden extenderse hasta 1051 días, capturando situaciones como lesiones o pausas en la carrera profesional.

3.3. Preprocesamiento de los datos

Una vez completada la extracción y el análisis descriptivo de los datos, resulta fundamental aplicar una serie de transformaciones que preparen la información para su utilización en un modelo de aprendizaje automático. El preprocesamiento constituye una etapa crítica que determina en gran medida la efectividad del modelo predictivo, ya que estos modelos son especialmente sensibles a la escala, y al formato de los datos de entrada.

3.3.1. Normalización de probabilidades de las casas de apuestas

En primer lugar, las probabilidades extraídas de las casas de apuestas requieren normalización antes de su anexión al modelo, ya que incorporan el margen de beneficio conocido como vigorish y no suman uno debido a esta medida. Para corregir esta distorsión, se implementa un proceso de normalización que ajusta las probabilidades para que su suma sea

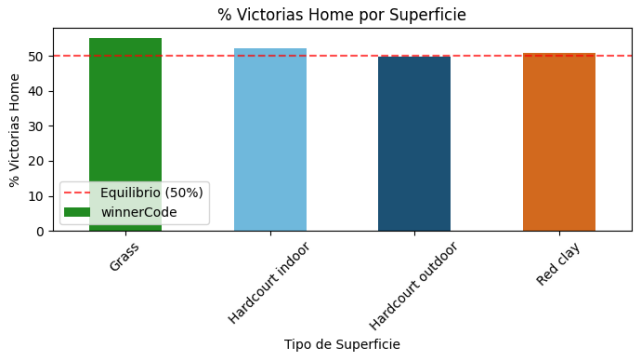

Figura 3.15: Porcentaje de victorias del jugador home según el tipo de superficie.

exactamente uno, permitiendo calcular el margen específico de cada partido. Esta transformación proporciona al modelo probabilidades que reflejan exclusivamente la expectativas del mercado sobre cada uno de los jugadores.

3.3.2. Transformación a diferencias

El paso más determinante del proceso de preprocesamiento es la transformación del DataSet "actual" a diferencias. Esta decisión responde a objetivos diferentes en cada conjunto de datos. Mientras que "actual" requiere capturar directamente la ventaja de un jugador sobre otro, tanto en variables de ranking como variables físicas, "previos" debe conservar las tendencias individuales de rendimiento de cada competidor a lo largo del tiempo.

Para el conjunto de datos "actual", las diferencias resultan más informativas que los valores absolutos para la predicción, ya que representan directamente la ventaja competitiva entre ambos jugadores. Esta aproximación facilita el trabajo al modelo, que no necesita establecer comparaciones entre variables equivalentes en ambos jugadores, sino que recibe directamente la información comparativa.

El enfoque general consiste en calcular la diferencia entre los valores de "homeז .ªway"para cada variable numérica continua, como el timestamp de la fecha de nacimiento, el peso y la altura. En esta codificación, valores más positivos indican ventaja del jugador home, mientras que valores más negativos señalan ventaja del jugador away.

Para los rankings, se implementa una transformación logarítmica que resuelve la problemática identificada en el análisis descriptivo, donde las diferencias absolutas reflejan efectos diferentes según el nivel competitivo. Esta transformación se calcula como `log(ranking-home/ranking_away)` y captura la realidad del tenis profesional: una diferencia de las mismas posiciones tiene un impacto competitivo muy diferente entre jugadores de élite (ventaja significativa) que entre jugadores de menor nivel (menor relevancia práctica). De este modo, el uso del logaritmo ajusta el peso competitivo de las diferencias de ranking de forma más realista. Por ejemplo, una confrontación entre jugadores de alto nivel (880 vs 820) produce un cociente logarítmico de 0.07, mientras que la misma diferencia absoluta entre jugadores de

menor nivel (140 vs 200) genera un logaritmo de -0.35, capturando así el impacto competitivo que representa una diferencia de 60 posiciones en niveles diferentes del ranking.

Por último, la diferencia de lateralidad se codifica mediante la concatenación de valores binarios (0 para zurdo, 1 para diestro) correspondientes a cada jugador, generando cuatro combinaciones posibles: 00 (ambos zurdos), 01 (home zurdo, away diestro), 10 (home diestro, away zurdo) y 11 (ambos diestros).

En contraste, el conjunto de datos "previos" mantiene la estructura original, conservando la información de cada jugador en formato Home/Away. Los registros históricos corresponden al mismo tenista dentro de cada bloque de datos, lo que permite mantener secuencias temporales específicas para cada uno.

3.3.3. Codificación de variables categóricas

El preprocesamiento continúa con la transformación específica de las variables categóricas para su correcta interpretación. Para el tipo de superficie, se implementa una codificación One-Hot que transforma la variable original "groundType" en cuatro variables binarias independientes: "groundType_Grass", "groundType_Hardcourt_indoor", "groundType_Hardcourt_outdoor" y "groundType_Red_clay". Cada variable toma el valor 1 cuando el partido se disputa en esa superficie y 0 en caso contrario.

La lateralidad sigue el mismo principio de codificación One-Hot, pero aplicado de manera diferenciada según el dataset. En "actual", se transforman las combinaciones de lateralidad en variables binarias tras la transformación a diferencias, mientras que en "previos", se mantienen las variables originales como binarias independientes, preservando la información individual de cada jugador requerida para el procesamiento secuencial.

La codificación One-Hot se emplea para transformar variables categóricas nominales en un formato que los algoritmos de aprendizaje automático puedan procesar correctamente. Esta técnica crea variables binarias independientes para cada categoría, asegurando que el modelo no interprete relaciones de orden inexistentes entre ellas. Por ejemplo, evita considerar que "hierba" es numéricamente superior a "tierra batida" simplemente por la asignación arbitraria de códigos numéricos. [25].

Adicionalmente, la variable "periodCount" se transforma en una variable binaria denominada "best_five", que toma el valor 1 cuando el partido se disputa al mejor de 5 sets y 0 cuando se juega al mejor de 3 sets.

3.3.4. Tratamiento de variables temporales

El siguiente paso consiste en transformar las variables temporales para optimizar su utilidad predictiva. En el DataSet "actual", se calcula directamente la diferencia de edad en años entre ambos competidores, mientras que en "previos", los timestamps de fecha de nacimiento se convierten en las edades individuales de los jugadores home y away. Para el tiempo

de inactividad, se calculan los días transcurridos desde el último partido de cada jugador basándose en la diferencia entre timestamps de partidos consecutivos. Adicionalmente, en el DataSet "actual" se incorpora un indicador temporal por semanas que permite al modelo identificar cada partido según la semana en que se disputó.

3.3.5. Estrategia de normalización y escalado

La elección del método de escalado se fundamenta en las características específicas de cada tipo de variable y su comportamiento esperado en el modelo predictivo. Para las variables físicas y temporales (altura, peso, edad, días sin jugar) se emplea StandardScaler, método que centra los datos en media cero y los escala a desviación estándar unitaria, preservando así las relaciones lineales naturales de estas variables.

En contraste, para las variables de rendimiento y clasificación (rankings, resultados, puntuaciones) se utiliza MinMaxScaler, que transforma los valores al rango [0,1] mediante una normalización basada en los valores mínimo y máximo de cada variable. Esta estrategia resulta más adecuada para variables con distribuciones asimétricas o rangos discretos como los rankings, que presentan una asimetría fuertemente negativa identificada en el análisis descriptivo.

La elección diferenciada de métodos de escalado se fundamenta en las características específicas de cada tipo de variable. StandardScaler preserva las relaciones lineales naturales y mantiene la información sobre la variabilidad original de las variables. En contraste, MinMaxScaler reescala los valores a un rango fijo [0,1], lo que lo hace más robusto ante valores atípicos y más apropiado para variables de rendimiento y clasificación que presentan distribuciones asimétricas [26].

3.3.6. Análisis final de las relaciones entre las variables

Una vez terminado el preprocesamiento, se realiza un análisis de las relaciones entre variables para validar la calidad de las transformaciones aplicadas. Como se observa en la figura 3.16, las variables de "actual" presentan correlaciones esperadas: la correlación de 0.74 entre diferencias de altura y peso refleja la relación física natural , mientras que la correlación moderada de 0.31 entre los rankings transformados confirma su relevancia sin generar redundancia.

El análisis del Factor de Inflación de la Varianza, que mide cuánto aumenta la varianza de una variable debido a su correlación con otras variables del modelo, complementa esta evaluación, mostrando en la figura 3.17 valores muy bajos (todos inferiores a 2.5) que confirman la ausencia de multicolinealidad entre las variables.

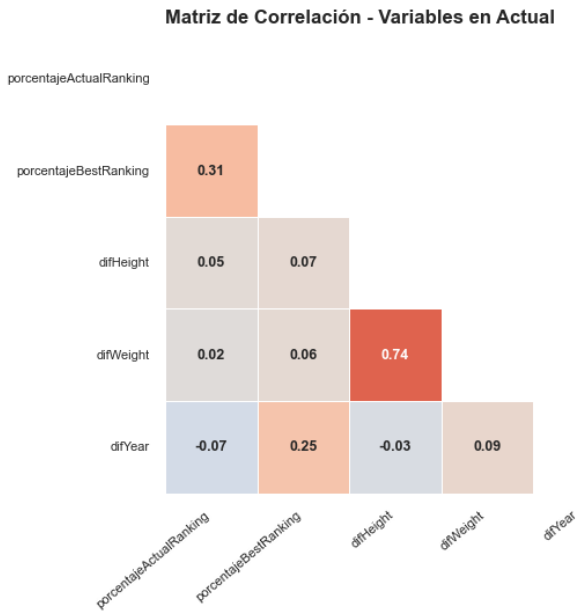

Figura 3.16: Matriz de correlación entre las variables numéricas del dataset "actual" tras las transformaciones aplicadas.

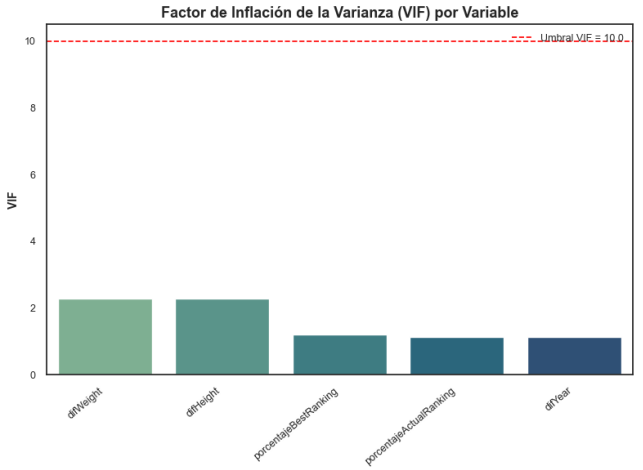

Figura 3.17: Factor de Inflación de la Varianza (VIF) para las variables del dataset "actual" tras la confirmación.

Estas métricas demuestran que las transformaciones aplicadas han resultado en conjuntos de datos estructurados y coherentes, donde el conjunto de datos "actual" contiene información predictiva organizada por diferencias, mientras que "previos" conserva la integridad temporal.

Capítulo 4

Diseño e implementación del modelo

Este capítulo presenta el diseño e implementación de la arquitectura híbrida desarrollada para la predicción de resultados en tenis profesional, combinando redes neuronales recurrentes para el procesamiento temporal con la integración de características contextuales del encuentro. Se detallan tanto la estructura del modelo como las estrategias de entrenamiento y validación empleadas para desarrollar los dos enfoques complementarios: clasificación binaria y regresión probabilística.

4.1. Arquitectura del modelo

El modelo desarrollado se estructura en tres componentes interconectados que procesan de manera diferenciada la información temporal y contextual disponible. En primer lugar, se emplean dos capas GRU independientes para procesar las secuencias temporales de cada jugador. Cada GRU recibe como entrada los últimos 50 partidos del jugador correspondiente, capturando así las tendencias de rendimiento, forma física y los patrones de juego específicos. De esta manera, el modelo aprende representaciones individualizadas para cada jugador.

El segundo componente consiste en una capa de proyección que procesa las características contextuales del partido actual. Esta información incluye las diferencias transformadas entre jugadores (ranking, características físicas, edad), y las condiciones del encuentro (superficie y formato).

La capa de proyección constituye una transformación lineal que duplica la dimensionalidad de las características del partido actual mediante combinaciones lineales de las variables originales. Esta operación mejora la representación contextual, permitiendo al modelo identificar interacciones más complejas entre las características del encuentro. Además, se ajusta

la dimensionalidad para compatibilizar con las salidas de las GRU, facilitando la posterior concatenación de la información temporal y contextual.

Finalmente, una red neuronal "fully connected" procesa la concatenación de las tres representaciones. La figura 4.1 ilustra la arquitectura completa del modelo, mostrando el flujo de datos desde las entradas hasta la salida final, así como las dimensiones específicas de cada componente.

Esta red comprende tres capas secuenciales con funciones de activación ReLU, que determinan la salida de cada neurona en función de su entrada y técnicas de regularización mediante dropout, finalizando en una capa de salida con activación sigmoide que transforma cualquier valor real al rango [0,1]. La función sigmoide se define matemáticamente como $\sigma(x) = 1/(1 + e^{(-x)})$, presentando una forma característica de "S"que mapea los valores de entrada al intervalo indicado, como se ilustra en la figura 4.2. En el modelo de clasificación, estas probabilidades se convierten a decisiones binarias aplicando un umbral de 0.5, mientras que en el modelo de regresión se utilizan directamente como estimaciones de probabilidad.

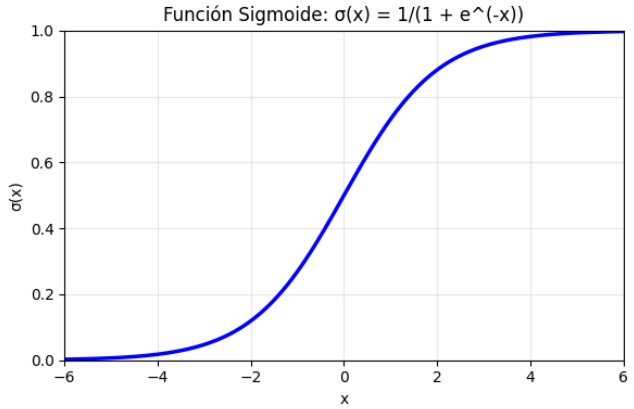

Figura 4.2: Función de activación sigmoide. La función mapea cualquier valor real de entrada al rango [0,1], con $\sigma(0) = 0.5$ como punto de inflexión.

4.2. Preparación de los datos

La preparación de datos para el modelo requiere una estructura que mantenga la integridad temporal y facilite el procesamiento por el modelo propuesto. La estrategia implementada combina la división temporal con transformaciones en cuanto a la forma necesarias para el funcionamiento de las redes neuronales recurrentes.

La división temporal de los datos emplea un enfoque de ventanas deslizantes que respeta la naturaleza temporal de los torneos de tenis. Una ventana deslizante es una técnica que organiza los datos en segmentos temporales consecutivos, los cuales se desplazan progresivamente a lo largo del tiempo. Cada ventana comprende 13 semanas consecutivas: 10 semanas

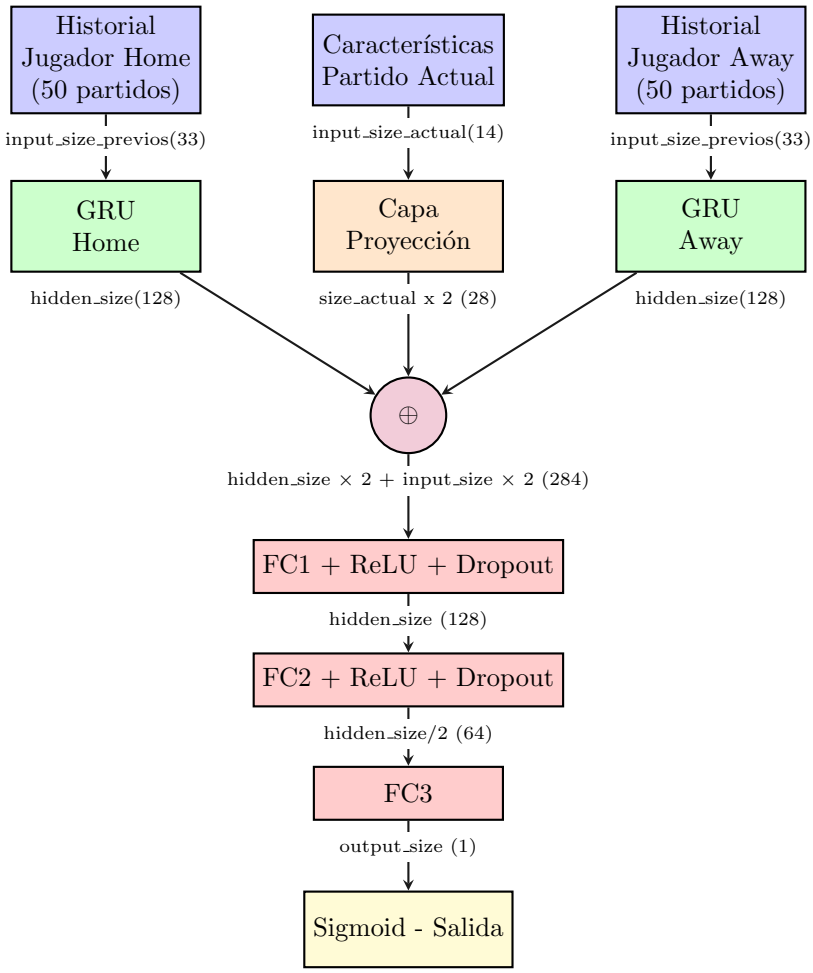

Figura 4.1: Arquitectura del modelo de predicción híbrido. El modelo procesa de forma independiente el historial de 50 partidos de cada jugador mediante capas GRU, integra las características del partido actual a través de una capa de proyección, y combina toda la información en una red neuronal fully connected para predecir la probabilidad de victoria.

destinadas al entrenamiento seguidas de 3 semanas para validación. Las ventanas avanzan con un paso de una semana, generando solapamiento entre períodos consecutivos. Este proceso genera un total de 143 ventanas temporales que cubren todo el período de estudio. La identificación temporal se basa en la variable `year_week_id`, que identifica cada semana de partidos en el conjunto "actual".

Para cada partido seleccionado en los conjuntos de entrenamiento o validación, se extraen exactamente 100 registros históricos del conjunto de datos "previos": 50 correspondientes al jugador local y 50 al jugador visitante, manteniendo la correspondencia 1:100 establecida anteriormente. Esta estructura asegura que cada partido analizado disponga de la misma cantidad de información histórica para ambos competidores.

Los datos históricos requieren una transformación dimensional específica para su procesamiento por las GRU. Los registros se reorganizan mediante operaciones de reshaping en tensores tridimensionales con dimensiones `[número_de_partidos, longitud_secuencia, número_características]`, donde el número de partidos varía según cada ventana temporal seleccionada, la longitud de secuencia corresponde a los 50 partidos históricos por jugador, y el número de características permanece fijo en 33 variables para cada conjuntos de datos. Esta restructuración convierte los datos en secuencias temporales que las redes recurrentes pueden procesar, ordenando la información cronológicamente desde el partido más antiguo hasta el más reciente.

Por otra parte, las características del partido actual se mantienen como vectores bidimensionales `[número_de_partidos, características_contextuales]`. Finalmente, todos los datos se convierten a tensores de PyTorch, estructura requerida para realizar las operaciones de la red neuronal, completando así la preparación necesaria para el entrenamiento del modelo.

4.3. Desarrollo de Modelos Duales

En función de la variable objetivo seleccionada, se desarrollan dos modelos complementarios que abordan perspectivas diferentes del problema de predicción en tenis profesional.

4.3.1. Modelo de Clasificación

La red neuronal de clasificación opera con el conjunto completo de 9.259 partidos del conjunto "actual" y sus correspondientes 925.900 registros históricos en "previos".

Este modelo predice la variable binaria winnerCode, que determina el resultado del encuentro: 0 indica victoria del jugador local y 1 victoria del visitante. Esta aproximación aborda directamente el problema fundamental de clasificación binaria planteado al inicio del proyecto.

La red utiliza la función de pérdida BCELoss (entropía cruzada binaria), que mide la discrepancia entre las predicciones del modelo y los resultados reales [27]. Esta función pe-

naliza logarítmicamente las predicciones incorrectas, mientras que favorece las predicciones correctas realizadas con certeza. Las métricas de evaluación para este modelo incluyen la precisión (accuracy), el área bajo la curva ROC (AUC), y la puntuación F1 (F1-score), proporcionando una evaluación completa de la capacidad discriminativa y del rendimiento del modelo

4.3.2. Modelo de Regresión

La red neuronal de regresión utiliza 8.466 partidos del conjunto "actual" y sus correspondientes 846.600 registros históricos en "previos", tras eliminar aquellos encuentros que carecen de información sobre las probabilidades de las casas de apuestas.

Este modelo predice la variable `probability_away`, que representa la probabilidad de victoria del jugador visitante según las expectativas del mercado. La selección de esta variable permite establecer una comparación directa con el modelo de clasificación, ya que ambos miden el mismo hecho. Cuando `winnerCode` toma el valor 1 (victoria del visitante), los partidos con `probability_away` más elevada se corresponden con resultados más próximos a 1.

Esta red utiliza la función de pérdida MSELoss (Mean Squared Error), que calcula el promedio de los errores cuadráticos entre las predicciones del modelo y los valores reales observados [27]. En el contexto de predicción de resultados deportivos, estas predicciones se comparan con las probabilidades reales del mercado, lo que permite evaluar la precisión del modelo en tareas de pronóstico. Las métricas de evaluación para este modelo incluyen el error absoluto medio (MAE), la raíz del error cuadrático medio (RMSE) y el coeficiente de determinación (R^2), proporcionando una evaluación plena de la calidad de las estimaciones.

4.4. Implementación de los modelos

Una vez definidas las arquitecturas y objetivos de ambos modelos, se procede a detallar la estrategia de implementación que abarca tanto el proceso de entrenamiento como el de validación. La implementación se desarrolla en Python utilizando el framework PyTorch, que proporciona las herramientas necesarias para la construcción y entrenamiento de redes neuronales recurrentes.

4.4.1. Entrenamiento

La estrategia de entrenamiento implementa un enfoque de ventanas deslizantes temporales que preserva la integridad cronológica de los eventos. Cada ventana temporal se procesa de manera independiente, asegurando que el modelo nunca acceda a información futura durante el proceso de aprendizaje.

El entrenamiento incorpora una técnica de "warm-up" entre ventanas consecutivas, donde cada nueva ventana inicializa los pesos del modelo con el mejor estado obtenido en la ventana anterior. Esta estrategia permite que el modelo se adapte gradualmente a los patrones evolutivos del deporte, manteniendo el conocimiento adquirido.

El proceso utiliza el optimizador "Adam" con una tasa de aprendizaje de 0.001, valor que proporciona un buen equilibrio entre velocidad de convergencia y estabilidad numérica, evitando grandes fluctuaciones la función de pérdida que podrían comprometer el entrenamiento de las capas recurrentes. La selección del optimizador"Adam" se justifica por su capacidad para manejar gradientes de diferentes magnitudes, característica especialmente relevante en modelos híbridos donde las capas recurrentes y fully connected presentan comportamientos diferentes en el gradiente [28].

La regularización se implementa mediante dropout con una tasa del 20 % en las capas fully connected. Esta técnica desactiva aleatoriamente el 20 % de neuronas durante el entrenamiento, forzando al modelo a desarrollar representaciones más robustas y distribuidas que mejoran su capacidad de generalización. Este porcentaje se selecciona considerando que las capas GRU ya incorporan mecanismos internos de regularización a través de sus compuertas, requiriendo un dropout menos agresivo que en arquitecturas feedforward.

Se establece un límite máximo de 100 épocas por ventana, aunque el entrenamiento puede detenerse anticipadamente mediante la implementación de "early stopping" para evitar el sobreajuste.

4.4.2. Validación

La validación del modelo se ejecuta de forma continua durante el entrenamiento, evaluando el rendimiento en el conjunto de prueba al finalizar cada época y monitorizando la evolución del aprendizaje.

El criterio de "early stopping" se basa exclusivamente en la pérdida de validación, estableciendo una mejora mínima de 0.00001 como umbral para determinar si existe progreso significativo entre épocas consecutivas. Cuando el modelo no logra superar este umbral durante 5 épocas consecutivas, el entrenamiento se detiene y se guarda el estado correspondiente a la mejor pérdida de validación obtenida.

Todas las métricas generadas durante el entrenamiento y validación se almacenan en conjuntos de datos organizado por ventana temporal y época, creando un registro completo que facilita el análisis posterior de los resultados obtenidos por el modelo.

Capítulo 5

Resultados obtenidos

Este capítulo presenta el análisis del rendimiento de los dos modelos desarrollados para la predicción de resultados en tenis profesional, comparando ambos y proponiendo un escenario de aplicación práctica.

5.1. Análisis del rendimiento del modelo de clasificación

El modelo de clasificación aborda el problema de predicción de la variable binaria que determina el resultado del encuentro. Para evaluar su rendimiento, resulta necesario analizar la precisión de sus predicciones y su estabilidad a lo largo del tiempo.

5.1.1. Inestabilidad temporal

El modelo de clasificación desarrollado presenta métricas promedio aparentemente moderadas a lo largo de las 153 ventanas temporales. La precisión (accuracy), que mide el porcentaje de predicciones correctas sobre el total de partidos, alcanza una media del 59.14 %. El área bajo la curva ROC (AUC) evalúa la capacidad del modelo para distinguir entre victorias del jugador local y visitante a través de diferentes umbrales de clasificación [27], se sitúa en el 63.22 %. Finalmente, el F1-score, que combina precisión y sensibilidad para reflejar el equilibrio entre falsos positivos y falsos negativos [27], alcanza un valor del 57.97 %.No obstante, estos valores promedio ocultan un problema fundamental de inestabilidad que compromete la viabilidad modelo.

Como se aprecia en la Figura 5.1, el modelo muestra una inestabilidad temporal elevada que constituye un gran defecto para su aplicación práctica. La precisión oscila de entre un 47 % y un 73 %, generando un rango de variabilidad muy elevado. Estas variaciones extrema se reproducen en todas las métricas: el AUC varía entre 51 % y 73 %, mientras que el F1-score

presenta oscilaciones similares, llegando a alcanzar valores cercanos al 47 % en determinados períodos.

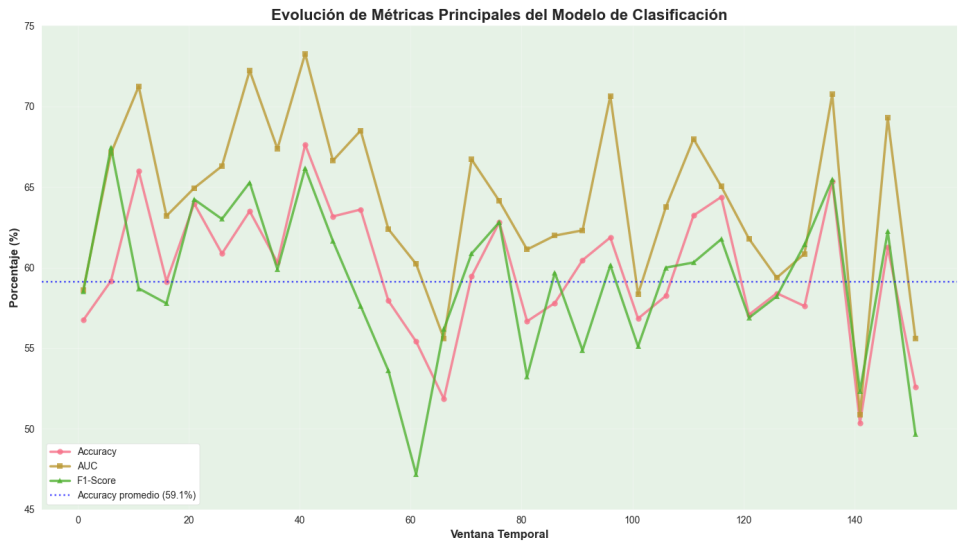

Figura 5.1: Evolución de las métricas principales del modelo de clasificación a lo largo de las ventanas temporales. Se observa inestabilidad temporal con fluctuaciones en todas las métricas.

Esta variabilidad temporal descarta la aplicación de este modelo, ya que la oscilación del rendimiento entre niveles cercanos al azar (47 %) y capacidad predictiva moderada (73 %) resulta inadecuada para la toma de decisiones.

Es importante reconocer que los eventos deportivos son impredecibles y que haya variabilidad en la precisión es esperable. El rendimiento del modelo depende significativamente de la ventana temporal y del nivel de dificultad de los partidos analizados, ya que no es comparable predecir rondas iniciales donde los favoritos suelen avanzar que enfrentamientos en fases finales entre rivales de alto nivel.

5.1.2. Problemas de generalización

El análisis de las curvas de pérdida durante el entrenamiento muestra la naturaleza de los problemas identificados. La Figura 5.2 refleja un patrón sobreajuste que explica la inestabilidad temporal observada en las métricas de rendimiento.

Mientras que la pérdida de entrenamiento (training loss) muestra un comportamiento descendente relativamente estable y predecible, la pérdida de validación (validation loss), que mide el error del modelo sobre el conjunto de prueba, presenta un comportamiento caracterizado por picos pronunciados en múltiples períodos temporales. Estos incrementos

son especialmente notables en las ventanas 50-70, 90-100 y 130-140, coincidiendo con los períodos de peor resultado en las métricas de rendimiento.

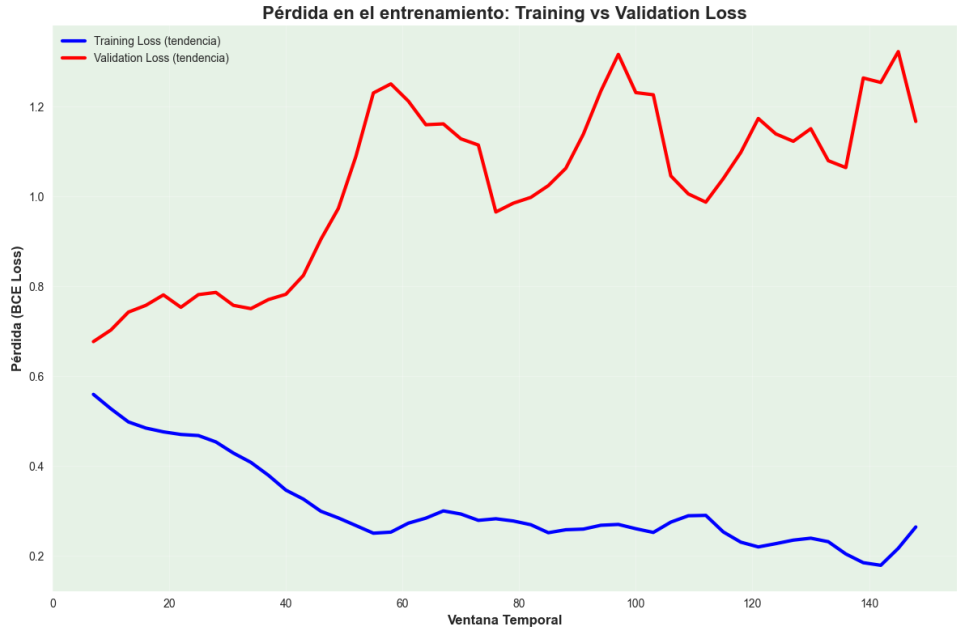

Figura 5.2: Comparación entre training loss y validation loss en el modelo de clasificación. La estabilidad descendente del training loss contrasta con la volatilidad del validation loss, revelando problemas de generalización.

Este patrón revela problemas graves de generalización que van más allá de un sobreajuste puntual. La repetición sistemática de estos episodios sugiere que el modelo presenta limitaciones fundamentales para aplicar lo aprendido durante el entrenamiento al enfrentarse a partidos con los que no ha sido entrenado. El contraste entre la estabilidad del training loss y los altibajos del validation loss indica que el modelo se adapta demasiado a cada ventana temporal específica. Además, la implementación de early stopping, aunque efectiva dentro de cada ventana individual, no logra resolver estos problemas de fondo que se manifiestan entre diferentes períodos temporales.

La exploración de diferentes configuraciones de hiperparámetros (variaciones en dropout, learning rate y criterios de early stopping) produjo resultados muy similares a los presentados, confirmando que estos problemas trascienden el ajuste de parámetros. Este comportamiento inestable revela que la capacidad del modelo resulta insuficiente para manejar la complejidad presente en los datos deportivos, donde cada ventana temporal presenta factores impredecibles. Por tanto, las limitaciones observadas corresponden a barreras del modelo para desarrollar representaciones que capturen la naturaleza dinámica del tenis, más que a problemas de configuración o entrenamiento.

5.2. Análisis del rendimiento del modelo de regresión

El modelo de regresión busca predecir las probabilidades de victoria que asignan las casas de apuestas antes de cada encuentro. Esta aproximación tiene especial relevancia para aplicaciones comerciales, ya que permite identificar discrepancias entre las predicciones del modelo y las expectativas del mercado, detectando oportunidades de beneficio en relación a las apuestas deportivas.

5.2.1. Evolución del modelo

El modelo de regresión presenta un comportamiento temporal claramente diferenciado en dos fases. Como se evidencia en la Figura 5.3, el modelo presenta dos etapas bien definidas que contrastan significativamente con la inestabilidad continuada que se observa en el modelo de clasificación.

Durante la fase inicial (aproximadamente ventanas 1-20), el modelo muestra un rendimiento bajo que se caracteriza por un coeficiente de determinación (R^2) próximo a cero, llegando incluso a valores negativos en algunos períodos. El (R^2), mide la proporción de variabilidad explicada por el modelo siendo 1 la predicción perfecta y 0 equivalente a predecir siempre el valor medio. Asimismo, el error absoluto medio (MAE), que cuantifica la media del valor absoluto de las diferencias entre las probabilidades predichas y las probabilidades reales [27], alcanza valores superiores al 15 %, indicando errores significativos en las estimaciones probabilísticas.

Sin embargo, a partir de la ventana 20, el modelo comienza a estabilizarse en valores superiores. El (R^2) aumenta progresivamente y se mantiene de forma consistente en valores superiores a 0.7, alcanzando un promedio estabilizado de 0.74. Esta métrica indica que el modelo logra explicar aproximadamente el 74 % de la variabilidad en las probabilidades de victoria, lo cual representa un rendimiento notable para predicciones deportivas. Es importante destacar que los eventos deportivos son altamente impredecibles y presentan gran variabilidad, por lo que un 75 % supone un buen resultado. Al mismo tiempo, el MAE se reduce y estabiliza en torno al 7.6 %, lo que supone errores promedio de menos de 0.08 en las estimaciones de probabilidad.

Por último, la métrica 1-RMSE (donde RMSE representa la raíz del error cuadrático medio) penaliza más severamente los errores elevados, se interpreta en una escala donde valores más cercanos a 1 indican un mejor rendimiento del modelo. Esta métrica se mantiene estable en torno al 0.9, confirmando la calidad de las predicciones tras la estabilización.

En el contexto del tenis profesional, estas métricas tienen un significado práctico concreto. Un R^2 de 0.74 indica que el modelo captura aproximadamente tres cuartas partes de los factores que determinan las probabilidades de victoria según el mercado de apuestas, incorporando exitosamente elementos como diferencias de ranking y superficie de juego. El MAE del 7.6 % implica que si las casas de apuestas asignan un 60 % de probabilidad a un jugador, el modelo típicamente predecirá entre 52 % y 68 %. Por su parte, el RMSE resulta especialmente relevante para identificar errores significativos en la estimación de favoritos

claros. Es decir, si una probabilidad real de 0.85 se predice como 0.4, este error impacta más en el RMSE que en el MAE.

Para ilustrar el comportamiento del modelo en casos específicos, se analiza el resultado obtenido en el enfrentamiento entre Nakashima y Majchrzak en Roland Garros 2022 (ventana 45). En este encuentro, el modelo predijo una probabilidad del 45.2 % de victoria para Majchrzak que coincidió exactamente con la probabilidad del mercado de apuestas (45.2 %). Este ejemplo demuestra la capacidad del modelo para capturar las expectativas del mercado en partidos equilibrados donde la diferencia de ranking era mínima (Nakashima 75º vs Majchrzak 79º), reflejando la incertidumbre de un enfrentamiento entre jugadores de nivel similar.

No obstante, el modelo presenta limitaciones ante eventos deportivos impredecibles. Un ejemplo representativo se observa en un enfrentamiento de primera ronda del Australian Open 2024 (ventana 105), donde el modelo predijo una probabilidad del 53.9 % de victoria para el jugador con peor ranking (99º), mientras que el mercado asignaba únicamente un 11.1 %, generando un error del 42.8 %. Este caso, refleja las discrepancias que pueden surgir entre las estimaciones del modelo y las expectativas del mercado en partidos particularmente difíciles de predecir.

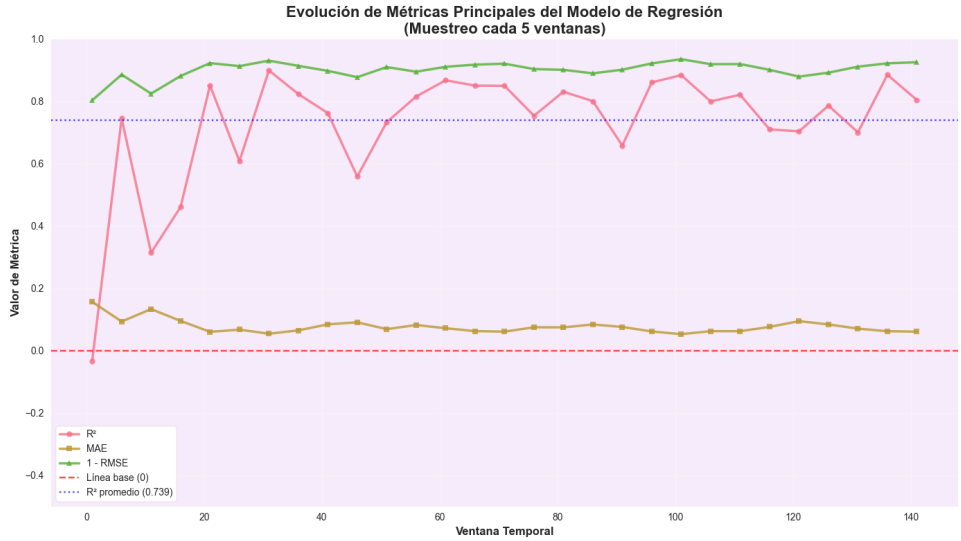

Figura 5.3: Evolución de las métricas principales del modelo de regresión a lo largo de las ventanas temporales. Se observa una fase inicial de adaptación seguida de estabilización en valores consistentes.

5.2.2. Calidad en las predicciones

Para comprender mejor estos resultados, es necesario examinar el comportamiento del modelo durante el proceso de entrenamiento. La Figura 5.4 muestra que, a diferencia del

modelo de clasificación, este modelo logra un proceso de convergencia estable tras el período inicial de adaptación.

La pérdida de entrenamiento (training loss) muestra un comportamiento descendente de manera continuada que se estabiliza en niveles bajos, mientras que la pérdida de validación (validation loss), que mide el error cuadrático medio sobre el conjunto de prueba, presenta inicialmente valores elevados que disminuyen progresivamente.

Aunque se observan fluctuaciones en la pérdida de validación a lo largo de las ventanas temporales, estas variaciones son esperables debido a que cada ventana temporal contiene partidos con características diferentes en cuanto a dificultad predictiva, como rondas iniciales versus finales o distintos tipos de superficie. No obstante, estas fluctuaciones son significativamente más moderadas que las observadas en el modelo de clasificación.

Este contraste con el modelo de clasificación resulta fundamental, ya que mientras la pérdida de validación del modelo de clasificación mostraba picos recurrentes que reflejaban problemas de generalización, el modelo de regresión muestra una convergencia efectiva una vez superado el período inicial.

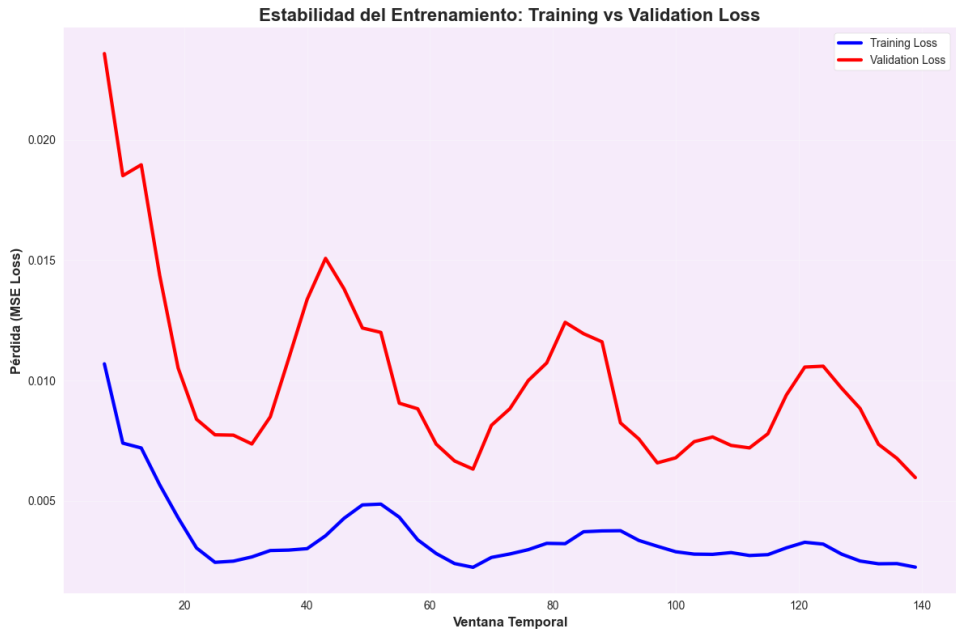

Figura 5.4: Comparación entre training loss y validation loss del modelo de regresión. Se observa convergencia estable tras el período inicial, en contraste a los problemas de generalización del modelo de clasificación.

La calidad en las predicciones resulta superior a la del modelo de clasificación. Con un R^2 consistente de 0.74, el modelo demuestra una capacidad sólida para capturar patrones en las probabilidades de victoria. Por otra parte, el MAE estabilizado del 7.6 % indica que

las predicciones del modelo presentan errores promedio 0.08 ountos en probabilidad, lo que supone un nivel de precisión aceptable para predicciones deportivas.

En definitiva, la estabilidad temporal constituye la diferencia fundamental respecto al modelo de clasificación. Mientras que este último mantiene una alta variabilidad e imprevisibilidad a lo largo de todas las ventanas temporales, el modelo de regresión demuestra capacidad para mantener un rendimiento consistente una vez superado el período inicial de adaptación.

A pesar de su estabilidad, este modelo presenta limitaciones técnicas específicas. Las fluctuaciones en las métricas entre ventanas sugieren posibles sesgos en la selección de variables contextuales, donde características como las diferencias de ranking pueden tener pesos desproporcionados según el período competitivo específico. Adicionalmente, la dependencia de las probabilidades de las casas de apuestas como variable objetivo introduce sesgo hacia las expectativas del mercado, limitando la capacidad del modelo para identificar patrones que el mercado no haya considerado previamente.

5.3. Análisis comparativo entre modelos

El análisis de ambos modelos en conjunto muestra diferencias claras en cuanto a su utilidad práctica. La fluctuación del rendimiento a lo largo de toda la ventana temporal supone un problema grave para el modelo de clasificación. Estas variaciones, junto con los problemas de generalización que presenta, hacen que sea imposible implementarlo en aplicaciones reales. Por otro lado, el modelo de regresión consigue estabilizarse en torno a la época 20 y mantener un rendimiento constante del 74 % en el R^2 con errores en torno al 7,6 %.

La diferencia fundamental está en cómo evolucionan los dos modelos con el tiempo. El modelo de clasificación mantiene sus problemas estructurales durante todo el período analizado, mientras que el modelo de regresión, aprende de los datos y se estabiliza después del período inicial. Esto hace que el modelo de regresión sea el candidato para su uso comercial.

5.4. Aplicabilidad práctica del modelo de regresión

La aplicación práctica del modelo de regresión se basaría en la identificación de discrepancias entre las probabilidades predichas y las implícitas en las cuotas de las casas de apuestas. El sistema funcionaría detectando automáticamente cuándo el modelo y el mercado valoran de manera diferente un partido, creando oportunidades de valor en las apuestas deportivas.

Un ejemplo concreto de oportunidad de valor se identifica en un encuentro de Roland Garros 2022 en tierra batida (ventana 44), donde el modelo predijo una probabilidad del 63.4 % de victoria, mientras que el mercado ofrecía únicamente un 20.8 % de probabilidad implícita (cuota 4.81).Esta discrepancia del 42.7 % indica una oportunidad excepcional donde el modelo detecta valor significativo. Para una apuesta de 100€ a cuota 4.81, el valor esperado se calcula considerando ambos escenarios posibles: si el jugador gana (probabilidad

según modelo: 63.4 %), el retorno total sería 481€ con una ganancia neta de 381€; si pierde (probabilidad: 36.6 %), la pérdida sería de 100€. El valor esperado resultante es: (63.4 % × 381€) + (36.6 % × (-100€)) = 204.95€, lo que representa una ganancia esperada del 204.9 % sobre la inversión inicial. En términos prácticos, esto significa que por cada euro apostado se espera ganar aproximadamente 2.05€, lo que supone una rentabilidad muy superior al riesgo asumido.

Inversamente, el sistema también identifica casos a evitar, como en un enfrentamiento del Australian Open 2024 (ventana 103), donde el modelo predijo solo un 22.8 % de probabilidad de victoria para el jugador visitante con peor ranking (142º) frente al favorito (ranking 114º), mientras que el mercado ofrecía un porcentaje de victoria del 60.2 % para este jugador (cuota 1.66). En este escenario, una apuesta de 100€ generaría un valor esperado negativo calculado como: (22.8 % × 66€) + (77.2 % × (-100€)) = -62.2€, muestra que se esperaría perder dinero a largo plazo, por lo que el sistema recomendaría evitar esta apuesta.

Para esta implementación práctica, sería necesario actualizar continuamente el modelo con datos recientes para mantener predicciones precisas en tiempo real, permitiendo así la identificación de discrepancias entre las predicciones del modelo y las expectativas del mercado.

Capítulo 6

Conclusiones y trabajo futuro

6.1. Evaluación del proyecto

6.1.1. Logros alcanzados en relación a los objetivos planteados

El objetivo relacionado con la automatización del proceso de extracción de datos ha sido alcanzado satisfactoriamente mediante el desarrollo de un sistema de Web Scraping que integra múltiples fuentes de información. La implementación logró recopilar 9.259 partidos válidos del circuito ATP, acompañados de 925.900 registros históricos que cubren las temporadas 2021-2024. La integración entre las dos fuentes de datos se consiguió resolver de manera efectiva a través de un sistema de correspondencia, alcanzando una precisión prácticamente perfecta en la identificación. El sistema mantiene la integridad temporal de los datos mediante la correspondencia 1:100 entre partidos actuales y registros históricos, garantizando que cada encuentro disponga de exactamente 50 partidos previos por jugador.

El segundo objetivo, correspondiente al desarrollo de un modelo predictivo preciso y estable a lo largo del tiempo, presenta resultados diferenciados según el enfoque adoptado. El modelo de regresión satisface los criterios establecidos, alcanzando una elevada estabilidad tras la fase inicial de adaptación con un coeficiente de determinación R^2 de 0.74 y un error absoluto medio del 7.6 %. En contraste, el modelo de clasificación no logra cumplir el requisito de estabilidad temporal, presentando oscilaciones entre un 47 % y un 73 % de precisión acompañadas de problemas de generalización que imposibilitan su aplicación práctica.

El tercer objetivo, centrado en el desarrollo de un sistema con potencial comercial, se cumple mediante el modelo de regresión. Este modelo es capaz de detectar discrepancias entre sus propias predicciones y las expectativas reflejadas en las cuotas del mercado de apuestas. En particular, permite evaluar cuándo las probabilidades reales de victoria, según el modelo, difieren de las probabilidades implícitas en las cuotas ofrecidas por las casas de apuestas, estableciendo así un mecanismo sistemático para detectar situaciones en las que el mercado podría estar subestimando o sobrestimando las posibilidades de un jugador.

6.1.2. Limitaciones identificadas

Durante el desarrollo del proyecto fueron surgiendo diversas limitaciones que afectaron a múltiples aspectos del sistema. En el ámbito de la extracción de datos, Sofascore implementa mecanismos de protección que eventualmente bloquean las peticiones automáticas cuando se alcanza un número determinado de consultas. Aunque esta limitación no representó un obstáculo significativo al aparecer en las etapas finales del proceso de extracción, sí que plantea un problema relevante de cara a la futura implementación de un sistema automatizado.

La dependencia de fuentes de datos externas constituye otra limitación relevante, ya que cualquier modificación en la estructura de las APIs puede comprometer la funcionalidad del sistema. Además, el sistema presenta limitaciones derivadas de la disponibilidad de información histórica, al establecer como requisito la existencia de 50 partidos previos por jugador. Esta restricción excluye del análisis encuentros que podrían aportar valor predictivo, especialmente aquellos que involucran jugadores emergentes.

Desde la perspectiva de modelado, el proyecto revela las limitaciones asociadas a la naturaleza impredecible del tenis profesional. La estabilidad del modelo de regresión presenta variaciones entre diferentes ventanas temporales, reflejando las fluctuaciones naturales en la dificultad predictiva en los encuentros. Aunque estas oscilaciones resultan esperables, suponen una limitación práctica para aplicaciones que requieren un rendimiento mantenido a lo largo del tiempo.

6.2. Trabajo futuro

El desarrollo futuro del sistema requiere la implementación de un pipeline automatizado que garantice la actualización continua de la base de datos. Para llevar el sistema a producción, resulta fundamental extender la cobertura temporal hasta 2025 y establecer un mecanismo de extracción que opere de forma programada, ejecutándose semanalmente tras la conclusión de cada jornada de torneos ATP. Este sistema debe incluir validaciones de calidad en los datos y la capacidad de adaptarse a modificaciones en las fuentes sin intervención manual.

La mejora del modelo constituye otro eje fundamental para el desarrollo futuro. El uso de arquitecturas avanzadas, como los modelos Transformer, podría aportar mejoras significativas en la precisión de las predicciones. Además, la incorporación de variables adicionales como la información sobre lesiones recientes podría mejorar considerablemente las predicciones.

Desde la perspectiva de la aplicabilidad comercial, resulta factible desarrollar un sistema de notificaciones inteligente que analice las predicciones del modelo en tiempo real, comparándolas con las cuotas ofrecidas por las casas de apuestas para notificar al usuario cuando se identifiquen oportunidades de valor.

Por último, la implementación de un dashboard interactivo permitiría visualizar métricas de rendimiento del modelo, evolución de las predicciones y análisis comparativo con el mercado, proporcionando una herramienta muy útil para la toma de decisiones estratégicas.

Apéndice A

Enlaces de interés

A.1. Enlace a github

El código completo del proyecto, incluyendo los archivos de extracción de datos, notebooks de análisis y implementación de los modelos de redes neuronales, se encuentra disponible en el repositorio de GitHub:

https://github.com/diegordr3/TFG_Estadistica_Diego_Rodriguez

El repositorio incluye documentación del proceso completo, desde la recolección de datos hasta la evaluación de los modelos, junto con las instrucciones necesarias para la ejecución de los archivos.

Bibliografía

[1] Daniel Glez-Peña, Anália Lourenço, Hugo López-Fernández, Miguel Reboiro-Jato, and Florentino Fdez-Riverola. Web scraping technologies in an api world. 2013.

[2] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth. *CRISP-DM 1.0: Step-by-step data mining guide*. SPSS Inc., 2000.

[3] https://github.com/erikhren. Fases de crispdm. https://camo.githubusercontent.com/b4602b0a5bc77ca2ab60969b57c3e1b1093ca385216bcdf1d8bb7e9ec1910b72/68747470733a2f2f6d69726f2e6d656469756d2e636f6d2f6d61782f313430302f312a30594b616952664f53525147746a4f6243516d7849412e706e67. Accedido: Junio 2025.

[4] Lasse Petteri Rouhiainen. *Inteligencia artificial: 101 cosas que debes saber hoy sobre nuestro futuro*. Alienta Editorial, Barcelona, primera edición edition, noviembre 2018.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[6] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. 5(4):115–133, 1943.

[7] Nurudeen Yemi Hussain. Deep learning architectures enabling sophisticated feature extraction and representation for complex data analysis. 2024.

[8] Marius-Constantin Popescu, Liliana Perescu-Popescu, Valentina E. Balas, and Nikos Mastorakis. Multilayer perceptron and neural networks. 2009.

[9] Ibomoiye Domor Mienye, Theo G. Swart, and George Obaido. Recurrent neural networks: A comprehensive review of architectures, variants, and applications. 15(9):517, 2024. Received: 21 July 2024; Revised: 22 August 2024; Accepted: 23 August 2024; Published: 25 August 2024.

[10] Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview. 2019. arXiv:1912.05911v1 [cs.LG].

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. 9(8):1735–1780, 1997.

[12] Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. 2014.

[13] M. Sipko and W. Knottenbelt. Machine learning for the prediction of professional tennis matches. Meng computing final year project, Imperial College London, 2015.

[14] A. De Seranno. Predicting tennis matches using machine learning. Master's dissertation, Ghent University, 2020. Supervisado por Prof. dr. ir. L. Martens & Prof. dr. ir. T. De Pessemier.

[15] Z. Fang, J. Zheng, and L. Hu. Tennis match win prediction based on rf and rnn. In *2024 International Conference on Data Science and Network Security (ICDSNS)*. Fujian Normal University, 2024.

[16] Sofascore. Acerca de: Información sobre el portal. https://corporate.sofascore.com/es/about/. Accedido: Junio 2025.

[17] Joshua Ofoeda, Richard Boateng, and John Effah. Application programming interface (api) research: A review of the past to inform the future. 15(3), jul 2019.

[18] José Manuel Rosa Moncayo. Qué es rest: Conoce su potencia. https://openwebinars.net/blog/que-es-rest-conoce-su-potencia/, May 2018. Blog post.

[19] Ecma International. ECMA-404: The JSON Data Interchange Standard. https://ecma-international.org/publications-and-standards/standards/ecma-404/, 2017. Accedido: Junio 2025.

[20] Anderson Sánchez. "solicitudes asíncronas (ajax) xhr api, fetch api y promesas. ¿qué es ajax?", August 2019. Accedido: Junio 2025.

[21] Cloudflare. ¿qué es la gestión de bots? https://www.cloudflare.com/es-es/learning/bots/what-is-bot-management/, 2025. Consultado el 21 de junio de 2025.

[22] International Organization for Standardization. Iso 3166 country codes. https://www.iso.org/iso-3166-country-codes.html, 2023. Accedido: Junio 2025.

[23] RapidFuzz Contributors. Rapidfuzz. https://github.com/rapidfuzz/RapidFuzz, 2023. Accedido: Junio 2025.

[24] Sport Bet World. How bookmakers generate profit: Industry secrets revealed. https://sportbetworld.com/blog/how-bookmakers-generate-profit-industry-secrets-revealed, 2023. Accedido: Junio 2025.

[25] X. Wang and Y. Fan. Effective methods of categorical data encoding for artificial intelligence algorithms. 2023.

[26] Md Manjurul Ahsan, M. A. Parvez Mahmud, Pritom Kumar Saha, Kishor Datta Gupta, and Zahed Siddique. Effect of data scaling methods on machine learning algorithms and model performance. 2021.

[27] Fernando Berzal. *Redes Neuronales & Deep Learning*. Editorial Universidad de Granada, 2018.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014.