PREDICCIÓN DEL RENDIMIENTO DE DEPORTISTAS MEDIANTE TÉCNICAS DE MACHINE LEARNING.

TRABAJO FIN DE GRADO
CURSO 2024-2025

AUTORES
FRANCISCO DANIEL ORTIZ DELGADO (NOTA: 7)
ALEJANDRO NAFRÍA MEDINA (NOTA: 7)

DIRECTOR
ISMAEL SAGREDO OLIVENZA

GRADO EN INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DE COMPUTADORES

FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

PREDICTION OF ATHLETE PERFORMANCE USING MACHINE LEARNING TECHNIQUES

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTHORS
FRANCISCO DANIEL ORTIZ DELGADO
ALEJANDRO NAFRÍA MEDINA

DIRECTOR
ISMAEL SAGREDO OLIVENZA

CONVOCATORIA: SEPTIEMBRE 2025

GRADO EN INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DE COMPUTADORES
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

1 DE SEPTIEMBRE DE 2025

DEDICATORIA

A nuestras familias, fuente inagotable de afecto y aliento, que nos enseñaron con su ejemplo a perseverar en cada desafío y a celebrar cada logro.

Al mundo del deporte, que nos mostró desde la infancia el valor del esfuerzo, la disciplina y el compañerismo. Dedicamos este trabajo a cada entrenamiento bajo la lluvia, a cada estrategia trazada en el banquillo y a las victorias compartidas en el vestuario.

A la Universidad Complutense de Madrid, escenario de crecimiento intelectual y humano, que, con sus aulas, sus pasillos y sus bibliotecas nos brindó el espacio para cuestionar, investigar y crear.

A todos los que, de una u otra forma, han participado en este camino: vuestra huella queda impresa en estas páginas.

A vosotros, va dedicada esta memoria.

AGRADECIMIENTOS

Este trabajo es fruto del esfuerzo y la colaboración de muchas personas a quienes deseamos expresar nuestro más sincero agradecimiento.

En primer lugar, a nuestro tutor, Ismael, por haber sido un faro en la tormenta: por su infinita paciencia, su dedicación y su acompañamiento desde el primer hasta el último día.

A nuestras familias, por su apoyo incondicional y su constante confianza, paciencia y comprensión, sin los cuales no habría sido posible llegar hasta aquí.

Nuestro agradecimiento también a toda la comunidad de la Universidad Complutense de Madrid, y muy especialmente a los profesores, ayudantes y personal de la Facultad de Informática, por los conocimientos y valores transmitidos dentro y fuera del aula a lo largo de estos años.

Finalmente, al Club Deportivo Oroquieta Espinillo, por facilitarnos el acceso y brindarnos su apoyo, y en particular a nuestros entrenadores David Novalbos Utrilla y Manuel Alejandro Liquiñano Mandillo "Lolo", así como a nuestros compañeros más cercanos —Mochón, Miguel, Pérez y Mario—, por compartir con nosotros esta experiencia y enriquecer este trabajo con su pasión y compañerismo.

A todos vosotros, ¡gracias!

RESUMEN

El presente Trabajo de Fin de Grado aborda el problema de la gestión de la fatiga en el fútbol, un factor determinante tanto en el rendimiento deportivo como en la prevención de lesiones. Para dar respuesta, se diseñó una aplicación que combina datos obtenidos mediante chalecos GPS con algoritmos de *machine learning*, con el fin de predecir en tiempo real el nivel de fatiga de los jugadores durante pausas de hidratación, tiempos muertos y descansos.

El proyecto incluyó un proceso exhaustivo de **adquisición, limpieza y análisis de datos**, mediante el uso de métricas fisiológicas extraídas de los dispositivos, su segmentación en intervalos cortos y la aplicación de técnicas de normalización y correlación para seleccionar las variables más relevantes. Estos datos constituyeron la base sobre la que se entrenaron y evaluaron distintos modelos predictivos.

Para asegurar la relevancia práctica de nuestra propuesta, el prototipo fue evaluado con entrenadores de distintas categorías:

- David Novalbos Utrilla, entrenador preferente.
- Manuel Alejandro Liquiñano Mandillo, técnico de Primera Autonómica de Madrid.
- Pablo López García, entrenador de Segunda División Española.
- Zuhaitz Astondoa Benítez, entrenador de fútbol base.

Los *feedbacks* obtenidos respaldan la utilidad de la aplicación para anticipar puntos críticos de fatiga y optimizar la gestión de la plantilla en tiempo real.

El código fuente puede consultarse en el repositorio de GitHub:

- **Backend**: https://github.com/DanielOrtizDelgado/backendtfg.git
- **Frontend**: https://github.com/DanielOrtizDelgado/frontendtfg.git

El dataset con los datos de los entrenamientos que hemos recopilado lo hemos publicado en Kaggle:

- https://www.kaggle.com/datasets/alejandronafria/football-fatigue-metrics-dsl

Con este enfoque integral —desde la adquisición y procesamiento de datos hasta la inteligencia predictiva y el soporte táctico—, nuestra memoria aporta un marco de trabajo novedoso para la aplicación del *machine learning* en el deporte de élite y de base, a la vez que sienta las bases para futuras mejoras como la ampliación de muestras, la integración con APIs oficiales y el despliegue en la nube.

Palabras clave

Machine learning, LaLiga Tech, Microsoft Azure, Feed-forward, ReLU, DSL, STATSports.

ABSTRACT

This Final Degree Project addresses the problem of fatigue management in football, a key factor both in performance and in injury prevention. To tackle this challenge, we designed an application that combines data collected from GPS vests with *machine learning* algorithms, in order to predict players' fatigue levels in real time during hydration breaks, time-outs, and half-time intervals.

The project included an extensive process of **data acquisition, cleaning, and analysis**, involving the extraction of physiological metrics from the devices, their segmentation into short intervals, and the application of normalization and correlation techniques to select the most relevant variables. These data served as the foundation for training and evaluating different predictive models.

To ensure the practical relevance of our proposal, the prototype was evaluated by coaches from different competitive levels:

- David Novalbos Utrilla, Preferente coach.
- Manuel Alejandro Liquiñano Mandillo, coach in the Primera Autonómica of Madrid.
- Pablo López García, coach in the Spanish Second Division.
- Zuhaitz Astondoa Benítez, grassroots football coach.

The feedback obtained confirmed the usefulness of the application to anticipate critical fatigue points and optimize squad management in real time.

The source code is available in the GitHub repository:

- **Backend**: https://github.com/DanielOrtizDelgado/backendtfg.git
- **Frontend**: https://github.com/DanielOrtizDelgado/frontendtfg.git

The dataset with training data has been published on Kaggle:

- https://www.kaggle.com/datasets/alejandronafria/football-fatigue-metrics-dsl

With this comprehensive approach —from data acquisition and processing to predictive intelligence and tactical support—, our dissertation provides a novel framework for the application of *machine learning* in both elite and grassroots sports, while laying the foundations for future improvements such as expanding the dataset, integrating official monitoring APIs, and deploying the platform in the cloud.

Keywords: Machine learning, football, fatigue prediction, sports performance, STATSports.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

Capítulo 1 - Introducción

En este capítulo presentamos el contexto y la motivación que dan sentido a nuestro TFG, establecemos los objetivos que perseguimos y detallamos el enfoque metodológico elegido. Nos inspiramos en iniciativas como **LaLiga Beyond Stats**, anunciadas en el portal de noticias de Microsoft en octubre de 2021, donde se describe cómo LaLiga Tech y Microsoft Azure emplean 19 cámaras ópticas y procesan más de 3,5 millones de puntos de datos por partido para ofrecer métricas avanzadas en tiempo real [1], así como en el propio portal web de Beyond Stats, que pone a disposición de aficionados y técnicos 21 indicadores tácticos y de rendimiento mediante un interfaz interactivo [2]. Siguiendo esta línea, nuestro proyecto aplicará técnicas de *machine learning* a datos fisiológicos recogidos en intervalos breves (pausas de hidratación, tiempos muertos y descansos) para predecir el nivel de fatiga de los futbolistas y proporcionar al cuerpo técnico información objetiva que optimice sustituciones y ajustes tácticos.

1.1 Motivación

La creciente exigencia en el fútbol de élite obliga a los equipos a exprimir al máximo el potencial de cada jugador para cumplir con los objetivos deportivos marcados. En un escenario donde los márgenes entre la victoria y la derrota son reducidos, disponer de información precisa y oportuna sobre el estado físico de los futbolistas se convierte en una ventaja competitiva decisiva.

Por otro lado, la labor de los entrenadores y preparadores físicos está sometida a una presión constante: deben planificar estrategias, gestionar la carga de trabajo y tomar decisiones críticas en cuestión de segundos, a menudo con datos incompletos o pocos intuitivos. Esta complejidad y urgencia dificultan la tarea de optimizar el rendimiento individual y colectivo a lo largo de la temporada.

Para realizar este trabajo hemos decidido apoyarnos en la plataforma STATSports una empresa fundada en 2007 en Newry, Irlanda del Norte, por Alan Clarke y Sean O'Connor. Su misión desde el inicio ha sido revolucionar el seguimiento y análisis del rendimiento deportivo mediante tecnología de precisión, ofreciendo herramientas en

tiempo real que ayudan a entrenadores, preparadores físicos y jugadores a tomar mejores decisiones.

La compañía se hizo un hueco en el mercado gracias a la combinación de GPS de alta precisión, sensores inerciales y software intuitivo, lo que permitió elevar los estándares del monitoreo en el deporte de élite. Muy pronto, equipos de fútbol, rugby y selecciones nacionales empezaron a confiar en sus soluciones.

Entre sus hitos más destacados se encuentran:

- Asociarse con clubes de la Premier League e Irlanda Rugby (2010–2014).
- Expansión internacional con oficinas en EE. UU. (2016), para cubrir deportes como la NFL, la NCAA o la MLS.
- El acuerdo con la Federación Inglesa de Fútbol (FA) en 2018, que integró su tecnología en todas las selecciones nacionales inglesas.
- El lanzamiento de la Apex Series en 2019, con un nivel de precisión centimétrico.
- La creación de soluciones más accesibles como Sonra Lite y Athlete Series, dirigidas también a universidades, ligas femeninas y deporte amateur.

Hoy, STATSports es considerado el líder mundial en monitorización de atletas y análisis de rendimiento por GPS, utilizado por clubes como Manchester City, Liverpool, Arsenal o Juventus, así como selecciones nacionales como Inglaterra, Argentina, Estados Unidos o Australia.

Con sedes en Irlanda del Norte, Londres, Chicago, Florida y Melbourne, la empresa ofrece soporte global y continúa innovando con plataformas basadas en la nube, modelos centrados en el jugador y herramientas de seguimiento para la rehabilitación. Actualmente, sus soluciones se aplican en miles de equipos de todo el mundo, desde academias de base hasta campeones del mundo.

1.2 Objetivos

El objetivo principal es el de desarrollar una aplicación de fácil acceso y consulta inmediata que, a partir de datos fisiológicos y de rendimiento recogidos en pequeños intervalos durante pausas de hidratación, tiempos muertos y descansos, prediga el nivel de fatiga de los jugadores y proporcione al cuerpo técnico la información objetiva necesaria para optimizar sustituciones y ajustes tácticos.

Otro objetivo es el de publicar el dataset resultante en un repositorio de acceso abierto como lo es **kaggle**[3]. Hasta la fecha existen muy pocos conjuntos de datos públicos que incluyan métricas de fatiga y rendimiento en el fútbol, lo cual dificulta la comparación de modelos y el avance colaborativo en este campo. Hacer disponible esta información facilitará futuras investigaciones y potenciará la reproducibilidad de los estudios en análisis deportivo.

1.3 Plan de trabajo

El desarrollo del TFG se ha estructurado siguiendo una planificación ágil basada en sprints semanales, en los que se han definido objetivos concretos (análisis de datos, elaboración de módulos, pruebas de modelos y documentación), con puntos de control internos cada semana para garantizar el avance. Además, cada dos semanas se han realizado revisiones formales con nuestro tutor Ismael y validaciones periódicas con nuestros entrenadores, lo que nos ha permitido iterar rápidamente, incorporar feedback de distintas perspectivas y asegurar tanto la calidad del trabajo como el cumplimiento de los plazos académicos. A continuación, detallamos las tareas principales por periodo:

Sprint 0 – Octubre 2024

Actividades: Investigación de plataformas y APIs deportivas (StatsPerform, Mediacoach, SportMonks, Sportradar, FootyStats, STATSports Apex Coach Series).

Entregable: Listado y resumen de fuentes y APIs seleccionadas.

Sprint 0 bis – Noviembre–Diciembre 2024

Actividades: Adquisición y configuración del chaleco sensorizado (conectividad y calibración).

Instalación y puesta en marcha de la aplicación de captura de datos.

Detección y resolución de incidencias: sincronización, formatos de salida y estabilidad de lectura.

Entregable: Informe de integración del chaleco y registro de problemas y soluciones.

Sprint 1 – 12–18 Feb 2025

Actividades: Presentación de los CSV iniciales (datos desde el chaleco).

Definición de sujetos de prueba (Daniel y Alejandro).

Definición de métricas DSL, HSR y HID.

Entregable: CSV base y documento de métricas a emplear.

Sprint 2 – 19–25 Feb 2025

Actividades: Revisión bibliográfica sobre métodos de distancia (Euclídea, Manhattan, Mahalanobis…).

Selección de algoritmos (k-NN, DecisionTree, RandomForest, Perceptrón Multicapa).

Entregable: Informe comparativo de algoritmos.

Sprint 3 – 26 Feb–4 Mar 2025

Actividades: Desarrollo de scripts xlsxAcsv.py y corregirCSV.py.

Cálculo automático de coeficientes de Pearson y rectas de regresión sobre DSL.

Entregable: Código de limpieza y análisis estadístico.

Sprint 4 – 5–11 Mar 2025

Actividades: Generación de matrices de covarianza y correlación.

Filtrado de variables con correlación ≥ 0,65 con DSL.

Entregable: Visualizaciones de correlaciones.

Sprint 5 – 12–18 Mar 2025

Actividades: Desglose de datos en segmentos (5×5 y 15×15).

Normalización exploratoria.

Primera prueba de Perceptrón Multicapa.

Entregable: Scripts de segmentación y resultados preliminares.

Sprint 6 – 19–25 Mar 2025

Actividades: Consolidación de CSV acumulado y normalizado.

Validación cruzada del Perceptrón Multicapa.

Entregable: CSV final y métricas de validación (MAE, R^2).

Sprint 7 – 26 Mar–1 Abr 2025

Actividades: Ajustes de segmentación y normalización.

Pruebas de Perceptrón y Random Forest.

Entregable: Comparativa de rendimiento de ambos modelos.

Sprint 8 – 2–8 Abr 2025

Actividades: Finalización del preprocesado (calorías, impactos).

Entrenamiento definitivo de Random Forest ($R^2 \approx 0{,}90$).

Entregable: Reporte de Random Forest con métricas finales.

Sprint 9 – 9–15 Abr 2025

Actividades: Generación de reportes de resultados por sujeto (Daniel Ortiz, Alejandro Nafría, conjunto).

Selección y configuración de XGBRegressor y XGBClassifier para DSL.

Entregable: Propuesta de modelos XGBoost.

Sprint 10 – 16–22 Abr 2025

Actividades: Actualización de CSV (conversión de horas a minutos, IDs, normalización de DSL).

Reentrenamiento y ajuste de modelos.

Entregable: CSV final revisado y métricas actualizadas.

Sprint 11 – 23–29 Abr 2025

Actividades: Análisis comparativo final (MLP vs. Random Forest vs. XGBoost).

Preparación de tablas y gráficos para la memoria.

Entregable: Borrador del capítulo de "Desarrollo de Modelos".

Sprint 12 – 30 Abr–6 May 2025

Actividades: Redacción y maquetación de la memoria.

Revisión interna de capítulos previos.

Entregable: Versión preliminar completa de la memoria.

Sprint 13 – 7–13 May 2025

Actividades: Desarrollo de la aplicación interactiva (Streamlit/Node.js).

Instalación de Node.js y MongoDB Compass.

Entregable: Prototipo de interfaz web.

Sprint 14 – 14–20 May 2025

Actividades: Implementación del endpoint de predicción (API XGBoost).

Pruebas de UX y ajustes de parámetros.

Entregable: Demo funcional de predicción en tiempo real.

Sprint 15 – 21–27 May 2025

Actividades: Pruebas de usuario con entrenadores.

Recogida de feedback y correcciones finales.

Entregable: Informe de usabilidad y plan de mejoras.

Sprint 16 – 28 May–3 Jun 2025

Actividades: Redacción de conclusiones y trabajo futuro.

Formateo final según normas académicas.

Entregable: Borrador de la memoria para entrega.

Sprint 17 – Jul–Ago 2025

Actividades: Finalización tanto de la memoria como de la plataforma.

Entregable: Memoria final y MVP de la plataforma disponible y dataset público.

Revisiones quincenales con el tutor (Ismael): 12 Feb, 26 Feb, 12 Mar, 26 Mar, 9 Abr, 23 Abr, 7 May, 21 May,15 Jul y 15 Ago – para presentar entregables, resolver dudas y ajustar el plan.

Capítulo 2 - Estado de la cuestión

En este capítulo analizaremos el marco teórico y práctico que sustenta nuestro TFG. Comenzaremos explorando la creciente competitividad en el fútbol (2.1) y el papel clave del preparador físico y el equipo técnico (2.2). A continuación, revisaremos proyectos relacionados en el ámbito de la analítica deportiva (2.3) y detallaremos la metodología seguida en nuestro estudio (2.4). Posteriormente, cerraremos con un repaso a los modelos de *machine learning* empleados (2.5).

2.1 La competitividad en el fútbol

La competitividad en el fútbol se entiende como la lucha constante por superar al rival en un deporte que, desde sus orígenes, ha evolucionado hacia una profesionalización extrema. Con la unificación de las reglas en Inglaterra en 1863, el fútbol pasó de ser un juego informal a un deporte organizado, y pocos años después la introducción de la FA Cup en 1871 y la creación de la Football League en 1888 sentaron las bases de la competición moderna. A lo largo del siglo XX, la internacionalización del fútbol con torneos como la Copa del Mundo en 1930 y la Copa de Europa en 1955 incrementó la exigencia táctica y estratégica, obligando a los equipos a adaptarse a estilos de juego muy diversos

Hoy en día, los márgenes entre la victoria y la derrota son mínimos, y cada detalle puede marcar la diferencia. La preparación física, la gestión de la plantilla, la capacidad táctica y el uso de tecnologías avanzadas se han convertido en factores decisivos. Los clubes invierten en sistemas de monitorización con GPS, acelerómetros o monitores de frecuencia cardíaca que permiten cuantificar el rendimiento con gran precisión y optimizar tanto entrenamientos como partidos. A esto se suma la presión económica y mediática que obliga a las entidades a mantener un nivel de competitividad constante para atraer talento, recursos y resultados.

En definitiva, la competitividad en el fútbol actual no se limita al talento individual de los jugadores, sino que abarca un ecosistema en el que confluyen innovación tecnológica, preparación física y mental, gestión de recursos y capacidad de

adaptación táctica. En este escenario, cualquier ventaja, por mínima que sea, puede convertirse en el factor decisivo que incline la balanza hacia la victoria.

2.2 El papel del preparador físico y del equipo técnico

El preparador físico es responsable de diseñar y supervisar programas de entrenamiento que optimicen la resistencia, la fuerza y la recuperación de los jugadores. Para ello, necesita conocer con precisión los niveles de fatiga, ya que esta variable influye directamente en la capacidad de rendimiento y en el riesgo de lesión. Un exceso de carga sin la recuperación adecuada puede provocar sobreentrenamiento, reducir la eficacia de las sesiones e incluso acortar la vida útil del deportista. Por el contrario, ajustar la intensidad en función de la fatiga real permite planificar entrenamientos más efectivos, personalizados al perfil de cada jugador.

El equipo técnico —entrenador, ayudantes y analistas— se apoya en estos datos para tomar decisiones tácticas y estratégicas durante la competición. Saber cuándo un futbolista está alcanzando un umbral de fatiga crítica puede determinar el momento idóneo para realizar una sustitución, planificar rotaciones a lo largo de la temporada o incluso modificar la estrategia de partido. De este modo, el conocimiento sobre la fatiga deja de ser una percepción subjetiva para convertirse en una herramienta objetiva que permite maximizar el rendimiento individual y colectivo.

Así, la estrecha colaboración entre preparador físico y cuerpo técnico convierte la información sobre la fatiga en un recurso estratégico fundamental, tanto para optimizar la calidad de los entrenamientos como para incrementar la competitividad en la competición.

2.3 Proyectos relacionados

En esta sección revisamos cinco iniciativas cuyo enfoque y resultados guardan afinidad con nuestro trabajo en monitorización y predicción de fatiga en fútbol. Para cada proyecto indicamos:

Título y autor(es), breve descripción de objetivos y metodología, principales hallazgos o aportaciones, significado de las técnicas clave (si fuese menester) y vínculo con nuestro TFG.

2.3.1 Modelización estadística en pruebas de esfuerzo y rendimiento deportivo (Matabuena Rodríguez, 2017) [4]

Este Trabajo Fin de Máster analizó datos de pruebas de esfuerzo realizadas a patinadores de élite y deportistas del Centro de Alto Rendimiento de Pontevedra, empleando variables fisiológicas como VO_2; volumen de oxígeno, frecuencia cardíaca y umbrales de rendimiento. El autor aplicó diferentes modelos estadísticos avanzados que permitieron predecir con gran precisión los resultados en competición, validando la utilidad de los tests indirectos para evaluar la condición física. Además, el estudio desarrolló una herramienta para estimar el consumo de oxígeno a partir de la potencia y la frecuencia cardíaca, más precisa que las existentes hasta ese momento. Este trabajo demuestra el valor de transformar datos fisiológicos en información práctica para el entrenamiento, algo que conecta directamente con nuestro TFG, donde también buscamos generar predicciones objetivas a partir de métricas registradas en contextos deportivos.

2.3.2 Análisis secuencial en el fútbol de rendimiento (Castellano Paulis & Hernández Mendo, 2000) [5]

Este estudio se centró en describir de manera sistemática la acción de juego en fútbol competitivo a través de un diseño observacional aplicado en partidos de la Copa del Mundo de Francia de 1998. Mediante la codificación y análisis de datos, los autores identificaron patrones de conducta que se repetían con una probabilidad significativamente mayor a la esperada por azar, revelando secuencias de juego y comportamientos tácticos característicos. La principal aportación de este trabajo fue

mostrar cómo la observación estructurada puede ayudar a comprender la dinámica del fútbol desde una perspectiva científica. Aunque se trata de un enfoque cualitativo y centrado en la acción táctica, guarda relación con nuestro proyecto en el interés común por detectar patrones que permitan anticipar el rendimiento, si bien en nuestro caso lo hacemos a partir de datos fisiológicos y modelos de *machine learning*.

2.3.3 Análisis de variables medidas en salto vertical relacionadas con el rendimiento deportivo y su aplicación al entrenamiento (Jiménez-Reyes, Cuadrado-Peñafiel & González-Badillo, 2011) [6]

En este trabajo se estudió la relación entre la capacidad de salto vertical y la aceleración en sprints de corta distancia en atletas de nivel nacional e internacional. Se midieron distintos tipos de saltos como Squat Jump o Countermovement Jump con cargas progresivas y se analizaron sus correlaciones con el rendimiento en sprints de 20 y 30 metros. Los resultados mostraron que una mayor capacidad de salto estaba asociada con mejores marcas en velocidad, validando el uso de estas pruebas como predictores del rendimiento deportivo. Este estudio es relevante porque evidencia cómo métricas fisiológicas específicas pueden anticipar el desempeño de un atleta, una lógica que también sustenta nuestro TFG al utilizar indicadores registrados por chalecos GPS para estimar el nivel de fatiga en futbolistas.

2.3.4 Utilización de coeficientes ofensivos para el análisis del rendimiento deportivo en el fútbol sala (De Bortoli et al., 2001) [7]

Este trabajo desarrolló un estudio longitudinal de nueve años sobre partidos oficiales de fútbol sala en Brasil, con el objetivo de cuantificar la producción ofensiva a través de diferentes índices estadísticos. Se establecieron métricas como la producción ofensiva total, la objetividad o la tasa de aprovechamiento, y se comprobó que los equipos ganadores obtenían valores significativamente superiores a los perdedores. Estos hallazgos mostraron que es posible predecir con mayor fiabilidad el rendimiento de un equipo no solo a partir de la cantidad de lanzamientos realizados, sino también de la calidad y la eficiencia de los mismos. La aportación conecta con nuestro proyecto en la medida en que ambos buscan traducir datos cuantitativos en indicadores

predictivos útiles, aunque en nuestro caso nos enfocamos en métricas fisiológicas individuales vinculadas a la fatiga.

2.3.5 Generación de una herramienta de Machine Learning para apoyar a deportistas (Juan Israel Baroffi González & Javier Parra González, 2024) [8]

Este Trabajo Fin de Grado desarrolló una aplicación de escritorio en Python que, a partir de datos obtenidos mediante la pulsera Fitbit, era capaz de predecir métricas como frecuencia cardíaca, calorías o pasos mediante distintos algoritmos de *machine learning*. Los autores evaluaron modelos como XGBoost, LightGBM, KNN, Random Forest y redes neuronales, seleccionando aquellos que ofrecían un mejor equilibrio entre precisión y velocidad de entrenamiento. El sistema resultante, llamado *HeartPred'it*, permitió a los usuarios visualizar sus métricas y predicciones en tiempo real, demostrando un error muy reducido y una gran aplicabilidad práctica en el entrenamiento personal. Este proyecto guarda una relación muy estrecha con nuestro TFG, ya que comparte la filosofía de utilizar *machine learning* para transformar métricas fisiológicas en predicciones útiles, aunque en nuestro caso trasladamos este planteamiento al fútbol con un enfoque específico en la fatiga de los jugadores.

2.4 Metodología del proyecto

Para el desarrollo de este TFG hemos seguido un enfoque iterativo y colaborativo, centrado en la comunicación fluida y el ajuste continuo de objetivos. Nuestra dinámica de trabajo se articula en tres tipos de reuniones:

1. **Reuniones internas semanales.**
   Cada semana, nos encontrábamos para compartir avances, depurar el código, intercambiar ideas sobre el diseño de la aplicación y planificar las tareas de la siguiente semana. Estas sesiones nos permitieron mantener un ritmo constante, detectar problemas con antelación y repartir el trabajo de forma equilibrada.
2. **Seguimiento quincenal con el tutor.**
   Cada quince días, celebrábamos una sesión de trabajo con nuestro tutor, Ismael. En estos encuentros presentábamos resultados intermedios (prototipos, métricas de rendimiento de los modelos, capturas de la interfaz…), resolvíamos

dudas metodológicas y recibíamos su retroalimentación experta para corregir el rumbo, refinar la estrategia y ajustar los entregables según los requisitos académicos.

3. **Evaluaciones bimensuales con el entrenador.**
   Además, cada dos semanas mantuvimos reuniones con el entrenador de Alejandro, quien evaluaba los avances desde la perspectiva de un experto en este ámbito. En ellas evaluaba la calidad de los entrenamientos y los resultados en los partidos. Estas reuniones están grabadas y guardadas.

Gracias a esta combinación de reuniones internas, tutorías académicas y validaciones deportivas, pudimos iterar de manera ágil y garantizar que el producto final integrase rigor científico, calidad de código y utilidad real para entrenadores y preparadores físicos.

2.5 Modelos empleados

A continuación, describimos brevemente los modelos de machine learning utilizados para la predicción del DLS, así como sus ventajas y desventajas.

Para evaluar la calidad de las predicciones generadas por los modelos de *machine learning*, utilizamos dos métricas de validación principales:

- **R^2 (Coeficiente de determinación)**: mide qué proporción de la variabilidad de la variable objetivo (DSL en nuestro caso) es explicada por el modelo. Su valor oscila entre 0 y 1, donde valores más cercanos a 1 indican un mayor poder explicativo. Un R^2 alto sugiere que el modelo capta adecuadamente las relaciones entre las variables independientes y la dependiente.
- **MAE (Mean Absolute Error, Error Absoluto Medio)**: cuantifica la diferencia media entre los valores predichos por el modelo y los valores reales observados. Al expresarse en las mismas unidades que la variable objetivo, el MAE permite interpretar fácilmente el error de predicción: cuanto menor es el valor del MAE, mayor es la precisión del modelo.

El uso conjunto de ambas métricas nos proporciona una visión equilibrada del rendimiento: el R^2 refleja la capacidad explicativa global del modelo, mientras que el MAE nos da una medida práctica del error promedio en las predicciones.

2.5.1 Perceptrón Multicapa (MLP)

El **Perceptrón Multicapa** es un tipo de red neuronal feed-forward compuesto por una capa de entrada, una o varias capas ocultas y una capa de salida. Cada neurona realiza una combinación lineal de sus entradas, seguida de una función de activación no lineal (en nuestro caso, **ReLU**) que introduce la capacidad de modelar relaciones complejas entre variables.

- **Ventajas**
    - Gran potencia expresiva: capaz de aproximar prácticamente cualquier función continua (teorema de aproximación universal).
    - Flexibilidad en el diseño: se pueden ajustar el número de capas y neuronas según la complejidad del problema.
- **Desventajas**
    - Requiere normalización cuidadosa de las entradas para converger de forma estable.
    - Propenso a sobreajuste (overfitting) si no se emplean técnicas de regularización (dropout, L2).
    - "Caja negra": difícil de interpretar internamente.

En nuestro TFG, el MLP se emplea como línea base para contrastar la capacidad de una arquitectura generalista de "caja negra" frente a modelos de árbol y de boosting.

2.5.2 Random Forest

El **Random Forest** es un ensamblado de **árboles de decisión** entrenados de forma independiente sobre submuestras aleatorias del conjunto de datos (*bagging*) y evaluados luego de forma agregada (votación o promedio).

Cada árbol se construye seleccionando en cada nodo la mejor división sobre un subconjunto aleatorio de características, lo que reduce la correlación entre árboles y refuerza la robustez frente a ruido.

- **Ventajas**
    - Muy resistente al overfitting gracias al promedio de múltiples árboles.
    - Pocas exigencias de preprocesado: no necesita normalización de variables ni codificación exhaustiva de categorías.
    - Facilita la estimación de importancia de variables.
- **Desventajas**
    - Menos preciso que modelos de boosting en conjuntos muy desequilibrados o con interacciones complejas.
    - Interpretabilidad global limitada: aunque cada árbol sea interpretable, el conjunto no lo es tanto.
    - Puede resultar lento y voluminoso en memoria cuando se usan muchos árboles.

En nuestro proyecto, Random Forest demostró un excelente equilibrio entre precisión y velocidad de entrenamiento, sirviendo tanto para predicción continua de fatiga como para clasificación en rangos de cansancio como se muestra en la ***Figura 1***.

2.5.3 XGBoost (Extreme Gradient Boosting)

XGBoost es una implementación altamente optimizada de **Gradient Boosting** sobre árboles de decisión. Funciona construyendo secuencialmente un conjunto de árboles donde cada nuevo árbol corrige los errores residuales del ensamble previo, minimizando de forma iterativa una función de pérdida regularizada.

- **Ventajas**
    - Alto rendimiento y escalabilidad: incorpora optimizaciones como histogramas, paralelización y manejo eficiente de datos dispersos.
    - Control fino de regularización (*L1*, *L2*, *learning rate*) para mitigar overfitting.
    - Funciona bien tanto en tareas de regresión continua (XGBRegressor) como de clasificación (XGBClassifier).
- **Desventajas**
    - Ajuste de hiperparámetros: el espacio es más amplio que en Random Forest (que efectivamente maneja menos parámetros "críticos"), aunque no necesariamente mayor que en un Perceptrón simple, donde learning rate, regularización y número de épocas también exigen búsqueda cuidadosa.
    - Mayor tiempo de entrenamiento que Random Forest en datasets muy grandes.
    - Requiere más recursos computacionales (CPU/GPU) para explotar toda su eficiencia.

En nuestro TFG, XGBoost ofreció la mejor precisión predictiva para la variable **DSL** (Dynamic Stress Load), alcanzando un R^2 cercano a 0,90.

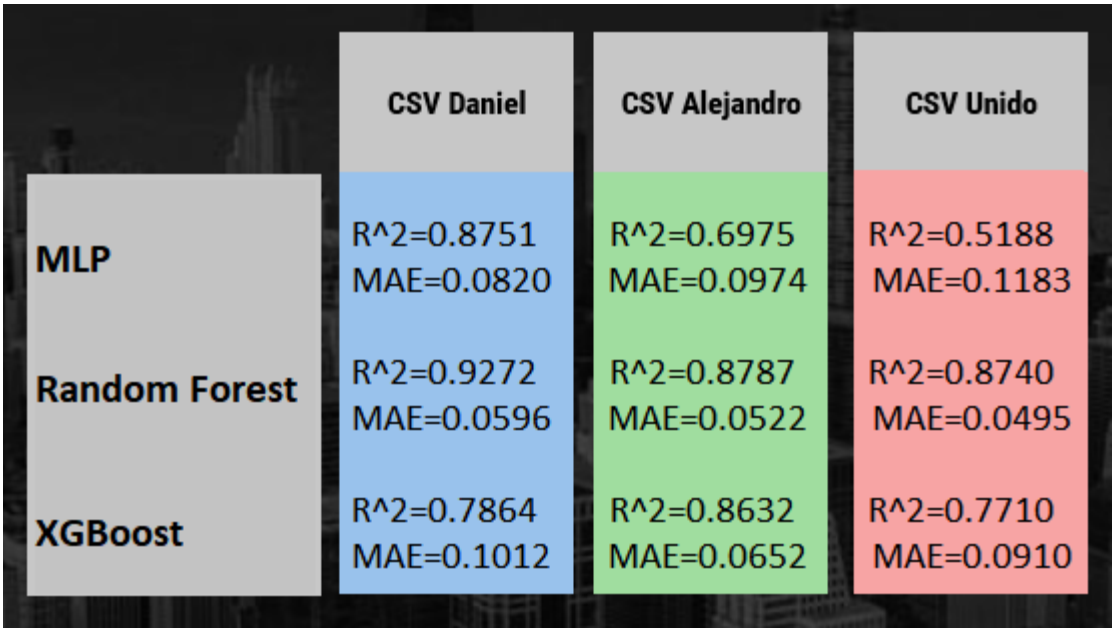


***Figura 1**- Cuadro resumen resultados*

Como se puede observar en la ***Figura 1*** los resultados muestran diferencias relevantes entre algoritmos y datasets. El modelo **Random Forest** fue el que alcanzó un rendimiento más consistente en los tres casos, con valores de R^2 cercanos o superiores a 0,87 y los MAE más bajos, lo que refleja un equilibrio óptimo entre capacidad explicativa y error medio. En cambio, el **MLP** ofreció resultados más irregulares según el dataset empleado, mientras que **XGBoost**, aunque competitivo, presentó una mayor variabilidad y un error superior en algunos escenarios. Estos datos refuerzan nuestra decisión de considerar a **Random Forest** como el modelo con mejor comportamiento global dentro del marco de nuestro TFG.

Comparativa algoritmos

Característica	MLP	Random Forest	XGBoost
Preprocesado	Requiere Normalización	Poco o nulo	Maneja valores perdidos
Riesgo de overfitting	Alto sin regularización	Bajo (bagging)	Bajo (boosting + regularización)
Interpretabilidad	Baja (caja negra)	Media (importancia de variables)	Media-alta (SHAP, gain plots)
Velocidad de entrenamiento	Media	Rápida	Variable (más lento)
Rendimiento en nuestro TFG	Base comparativa	Excelente balance	Mejor precisión (R^2 =0.90)

***Figura 2**- Comparativa algoritmos empleados*

Como se puede observar en la ***Figura 2***, en nuestro TFG se comprobó que el MLP servía como referencia inicial, el Random Forest ofrecía un excelente balance entre velocidad e interpretabilidad, y XGBoost, pese a su mayor complejidad computacional, fue el que alcanzó el mejor rendimiento predictivo sobre la variable DSL, con un R^2 cercano a 0,90.

Capítulo 3 - Adquisición y procesado de datos

En el fútbol moderno, uno de los principales problemas a los que se enfrentan entrenadores y preparadores físicos es conocer con precisión el nivel de fatiga de los jugadores. Esta variable resulta clave porque condiciona el rendimiento en el terreno de juego y determina el riesgo de lesión, pero no puede medirse de forma directa. Para poder estimarla con rigor es necesario registrar indicadores fisiológicos y de rendimiento a través de dispositivos especializados, y posteriormente procesar estos datos para transformarlos en información útil. En este capítulo describimos el procedimiento seguido para la adquisición y preparación de los datos empleados en nuestro proyecto, desde la captura inicial con el chaleco GPS hasta la obtención de un dataset normalizado listo para alimentar los modelos de *machine learning*.

En el marco de este Trabajo de Fin de Grado, adquirimos un chaleco GPS de STATSports para la recogida de datos. Gracias a esta tecnología, utilizada en el deporte profesional, fue posible obtener métricas precisas y en tiempo real sobre la actividad de los jugadores, constituyendo la base para nuestro análisis y experimentación.

Como muestra la **Figura 3** que está a continuación, recientemente, la plataforma STATSports ha lanzado una funcionalidad de pago basada en inteligencia artificial que, de forma muy similar a nuestro proyecto, predice el nivel de fatiga de los jugadores en tiempo real. Esta novedad pone de manifiesto el interés creciente por soluciones de este tipo, al tiempo que subraya el valor de contar con alternativas accesibles y adaptadas a las necesidades de clubes y cuerpos técnicos con recursos limitados.

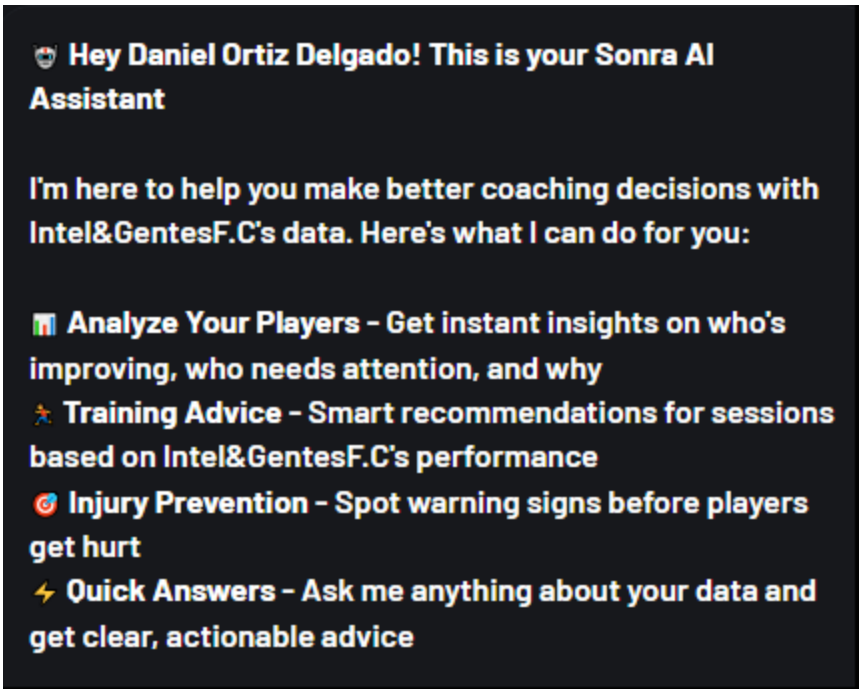


***Figura 3**-Publicidad STATSports futura integración de IA*

Para capturar la información cruda de cada sesión, nosotros, como deportistas, nos poníamos el chaleco con el dispositivo a la espalda. Tras presionar el botón de encendido y esperar a oír tres pitidos que confirman el inicio de la sesión. Una vez concluido el entrenamiento o partido, enlazamos el chaleco con nuestra aplicación móvil para volcar los datos.

A continuación, entramos, desde nuestro ordenador, en el portal de Sonra Lite (https://sonralite.statsports.com/) con nuestra cuenta y accedemos al menú **Calendario**.

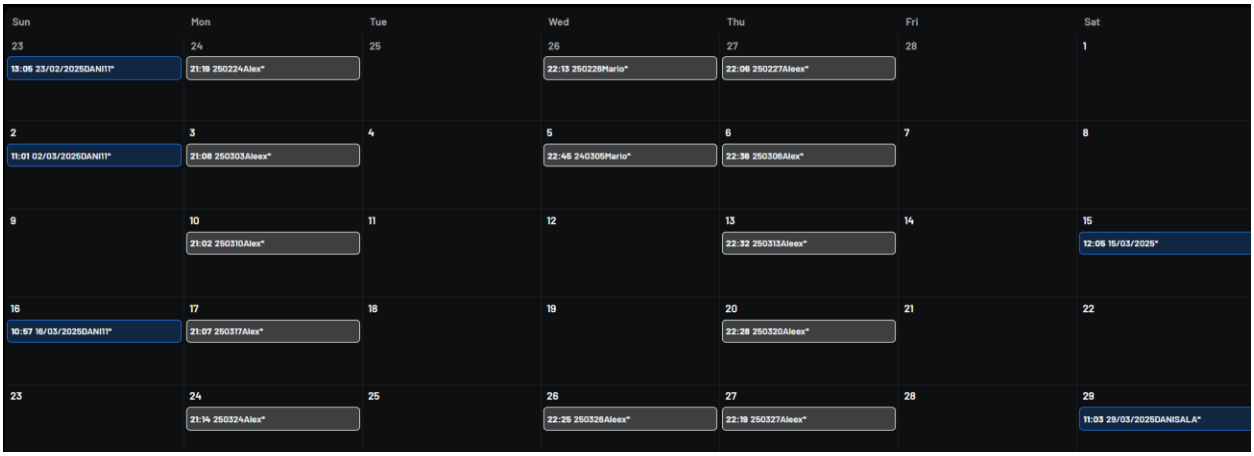


***Figura 4**- Calendario Apex Coach Series*

Seleccionamos la sesión correspondiente desde la vista que se muestra en la ***Figura 4***, añadimos el nombre del jugador y la fecha, y descargamos el fichero en formato CSV como puede verse en la ***Figura 5***.

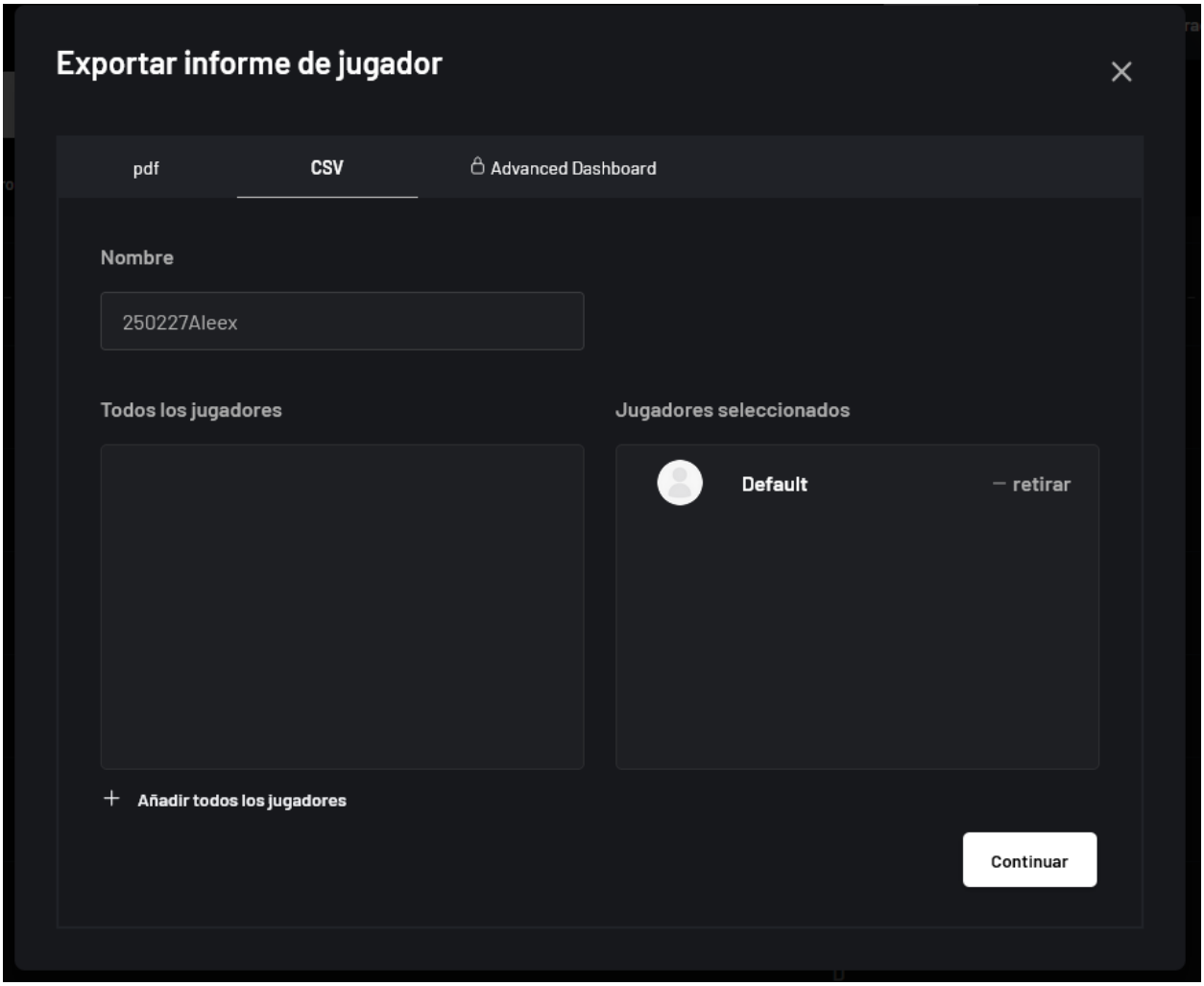


***Figura 5**- Exportar datos*

El chaleco GPS de **STATSports** nos proporciona un conjunto amplio de métricas relacionadas con el rendimiento físico del jugador. A continuación, detallamos las principales variables recogidas, junto con una breve explicación de cada una:

- **Distancia total**: recorrido total en metros durante la sesión.
- **Correr a alta velocidad (HSR, High-Speed Running)**: distancia recorrida a velocidades superiores a un umbral predefinido (≈ 19,8 km/h).
- **Distancia de alta intensidad (HID, High Intensity Distance)**: suma de los metros recorridos a velocidades muy elevadas o en esfuerzos de gran exigencia.
- **Distancia de sprint**: metros acumulados corriendo por encima del umbral de sprint (> 25,2 km/h).

- **Distancia por minuto**: media de metros recorridos por minuto de sesión.
- **HSR por minuto**: metros a alta velocidad por minuto de juego.
- **HID por minuto**: metros de alta intensidad por minuto.
- **Distancia de sprint por minuto**: media de metros en sprint por minuto.
- **Número de sprints**: cantidad de esfuerzos explosivos de alta velocidad detectados.
- **Velocidad máxima**: pico de velocidad alcanzado por el jugador en la sesión.
- **Impactos**: número de colisiones o contactos físicos registrados.
- **Aceleraciones**: total de aceleraciones significativas (≥ 3 m/s²).
- **Desaceleraciones**: total de desaceleraciones bruscas (≤ -3 m/s²).
- **Calorías**: gasto energético estimado a partir de la carga externa.
- **Step Balance (L/R)**: simetría entre pierna izquierda (L) y pierna derecha (R) durante la carrera.
- **DSL (Dynamic Stress Load)**: indicador agregado de la carga de esfuerzo basado en los impactos de los pies contra el suelo y aceleraciones, asociado al riesgo de fatiga y lesión.

Estas métricas brutas constituyen la base sobre la que se construye nuestro dataset. A partir de ellas se seleccionan las más relevantes, se segmentan en intervalos temporales y se someten a un proceso de limpieza y normalización, tal y como se describe a continuación.

Dentro de estas métricas, el **DSL (Dynamic Stress Load)** se definió como variable objetivo de nuestro proyecto, ya que representa de manera integrada el nivel de carga soportado por el jugador y su relación con la fatiga.

Conviene señalar que el cálculo del DSL es un **proceso interno del dispositivo** cuyo origen exacto desconocemos: no disponemos de la fórmula oficial ni de un baremo que establezca de manera objetiva el nivel de fatiga asociado a cada valor. Sin embargo,

hemos comprobado que los resultados registrados **coinciden con las sensaciones observadas en partidos y entrenamientos**, lo que refuerza su validez práctica. En nuestro caso, la única información disponible proviene del histórico de valores exportados en los ficheros CSV, sobre los que se ha construido el dataset.

Para determinar qué variables resultaban más relevantes como predictores del DSL, aplicamos un análisis exploratorio apoyado en:

- **Coeficiente de correlación de Pearson**: medida cuantitativa de la fuerza y dirección de la relación entre dos variables. (Apéndice D)
- **Matriz de correlación**: visualización de las relaciones lineales entre todas las métricas registradas. (Apéndice E)

Este análisis nos permitió identificar qué métricas presentaban mayor grado de asociación con el DSL y, por tanto, debían conservarse en el dataset para los modelos de *machine learning*.

Para limpiar el dataset, eliminamos las columnas irrelevantes: *session type*, *fc max*, *fc media* y *horas en zona roja*.
Seguidamente, usamos la aplicación móvil para desglosar los datos en intervalos de 5 minutos, extrayendo las siguientes métricas:

- Distancia total
- Correr a alta velocidad (HSR)
- Distancia de alta intensidad (HID)
- Distancia por minuto

Una vez obtenido el desglose cada 5 minutos, agrupamos estas ventanas de tres en tres para generar tramos de 15 minutos. Este proceso de agrupación nos permite suavizar la variabilidad y aportar más información al modelo, dándole un mayor contexto para extrapolar lo que ha pasado durante el entrenamiento o partido.

Finalmente, normalizamos todas las variables (estandarización *z-score* o escalado *min-max*; que es el empleado por nostros) y exportamos el CSV resultante, listo para su ingestión por los algoritmos de *machine learning* seleccionados.

Capítulo 4 - Aplicación y su funcionamiento

En este capítulo se describe en detalle el desarrollo de la plataforma orientada a la gestión de métricas deportivas y a la predicción de la carga de fatiga (DSL) tanto en jugadores de fútbol 11 y fútbol sala como su modo de funcionamiento.

4.1 Definición de requerimientos

Tras la preparación de los conjuntos de datos y la fase de selección de modelos, el siguiente paso consiste en el diseño de una plataforma que permita visualizar la información más relevante recopilada durante la temporada y mostrar el funcionamiento de las predicciones en un entorno realista donde poder mostrar la utilidad de hacer un seguimiento del estado de los jugadores como poder realizar ajustes, ya sea durante un partido o un entrenamiento, según estas predicciones. Los requerimientos de esta plataforma no consisten únicamente en comprobar la validez de los modelos predictivos sino en acercar la herramienta a aquellos usuarios finales que podrían beneficiarse del uso de esta.

Para el correcto desarrollo de la plataforma resulta necesario identificar los actores principales, las funcionalidades clave que debe cubrir el sistema y el alcance inicial de esta primera versión.

4.1.1 Actores Principales

Las predicciones que se pueden obtener actualmente están orientadas a que un **entrenador** pueda predecir el DSL o fatiga de un **jugador** de cara al final de un entrenamiento o de un partido, ya sea fútbol sala o fútbol 11. Teniendo esto en mente aparecen dos actores principales donde poner el foco. Estos son:

- **Entrenadores**: son los principales beneficiarios del sistema, ya que podrán consultar la información de todo el equipo, analizar las métricas de sus jugadores de forma individual y acceder a las predicciones de fatiga para planificar sesiones, alineaciones y sustituciones.

- **Jugadores**: en esta primera versión su papel es secundario, limitado a la visualización de sus propias métricas registradas en los entrenamientos y/o partidos en los que hayan participado.

4.1.2 Funcionalidades clave

Con el fin de validar la utilidad de los modelos y construir una plataforma con unos requisitos mínimos tanto de calidad como de contenido, se han definido un conjunto de funcionalidades mínimas:

- **Gestión de usuarios**: el sistema permite registrarse tanto a la figura de entrenador como la de jugador. Esto queda reflejado en la ***Figura 11*** y ***Figura 12***.
    - El **entrenador** debe proporcionar sus datos de usuario (nombre, correo y contraseña), crear un equipo asignándole nombre y una contraseña que deberá compartir con sus jugadores para que estos puedan inscribirse como jugadores del club, e indicar el tipo de deporte (fútbol 11, fútbol sala o mixto).
    - El **jugador** debe introducir sus datos de usuario junto con su posición en el campo y unirse a un equipo existente mediante el nombre y contraseña definidos por su entrenador.
- **Visualización de métricas**:
    - El **entrenador** puede consultar métricas relevantes de sus jugadores (distancia total, distancia a alta intensidad, velocidad máxima, impactos, DSL, etc.) Este dispondrá de una lista con la que podrá acceder a las métricas de cada uno de sus jugadores, como se muestra en la ***Figura 18***, la ***Figura 19*** y la ***Figura 20***.
    - El **jugador** únicamente puede consultar sus propias métricas a lo largo de las sesiones registradas plasmadas en la ***Figura 19*** y ***Figura 20***.
- **Calendario de sesiones**: el entrenador dispone de un calendario donde visualizar entrenamientos y partidos ya realizados. ***Figura 21***.

- **Predicción de DSL**: el **entrenador** puede predecir la fatiga de un jugador. Para esto necesitará adjuntar un archivo CSV o XLSX con un formato concreto y el conjunto de datos ya previamente montado; ***Figura 14*** y ***Figura 15,*** seleccionar tanto el modelo de predicción con el que trabajar como al jugador del cual se han obtenido los datos y elegir si la predicción se realizará utilizando el modelo con los datos del jugador en cuestión o el modelo con los datos de todo el equipo; ***Figura 16***.

  Tras este proceso el sistema devuelve el valor de DSL predicho, un porcentaje de fiabilidad del modelo utilizado que ayudará a saber qué tan acertada es la predicción y una interpretación textual del estado físico esperado del jugador al finalizar la sesión; ***Figura 17***. Esta interpretación se obtendrá a partir de un escalado del DSL de todas las sesiones (bien del jugador o bien del equipo) de 0 a 1, siendo 0 el valor más bajo y 1 el valor más alto registrados. Según el cuartil en el que se encuentre la predicción del DSL el jugador podrá estar: fresco, un poco fatigado, fatigado o exhausto.

4.1.3 Alcance Inicial

Dado que se trata de una primera versión, el alcance de la plataforma presenta ciertas limitaciones que conviene especificar:

- **Escasez de datos**: la recopilación se ha realizado únicamente con un chaleco sensorizado y alternando entre dos personas, lo que restringe el tamaño del conjunto de datos. Como alternativa para mitigar el desequilibrio y la falta de muestra suficiente en determinadas franjas de fatiga, se aplicará SMOTE (Synthetic Minority Oversampling Technique), una técnica de sobremuestreo que genera ejemplos sintéticos a partir de las observaciones minoritarias. De esta manera, se incrementaría artificialmente el número de instancias en las regiones de mayor interés —por ejemplo, estados de fatiga extrema poco frecuentes—.
- **Persistencia estática**: en esta primera versión no existirá la posibilidad de añadir nuevas sesiones, ni datos de los jugadores presentes en ellas, directamente desde la interfaz las predicciones se realizan a partir de archivos CSV/XLSX previamente

preparados, con datos organizados y segmentados para el entrenamiento y la inferencia de los modelos.

- **Ámbito de despliegue**: la aplicación no se encuentra desplegada en la nube; su funcionamiento se limita al entorno local de desarrollo.

En definitiva, el alcance inicial busca demostrar la viabilidad del sistema y ofrecer una primera versión funcional que permita validar tanto la utilidad de las métricas como la aplicabilidad de las predicciones en contextos de fútbol 11 y fútbol sala.

4.2 Diseño inicial de la aplicación

En este apartado vamos a mostrar brevemente el diseño de la interfaz de la aplicación y su funcionalidad básica, con el objetivo de describir las diferentes posibilidades que esta dispone.

***Figura 6**-Registro Plataforma*

La ***Figura 6*** muestra la **pantalla de registro y acceso a la plataforma Khomün**. En ella se incluye un mensaje de bienvenida que destaca el propósito de la aplicación —predecir en tiempo real la fatiga de los jugadores para optimizar las sustituciones—, junto con el formulario de inicio de sesión mediante correo electrónico y contraseña.

Además, se ofrece la opción de recuperación de credenciales y el registro diferenciado de entrenadores y jugadores, facilitando un acceso personalizado según el rol dentro del sistema.

***Figura 7**- Registro Entrenador*

En la ***Figura 6*** podemos ver el **diseño inicial de la aplicación Khomün**, con el logotipo en la parte superior y la pantalla de acceso principal. Incluye un mensaje de bienvenida que resume la propuesta de valor de la plataforma —predecir en tiempo real la fatiga de los jugadores— y un formulario para iniciar sesión o registrarse como nuevo usuario.

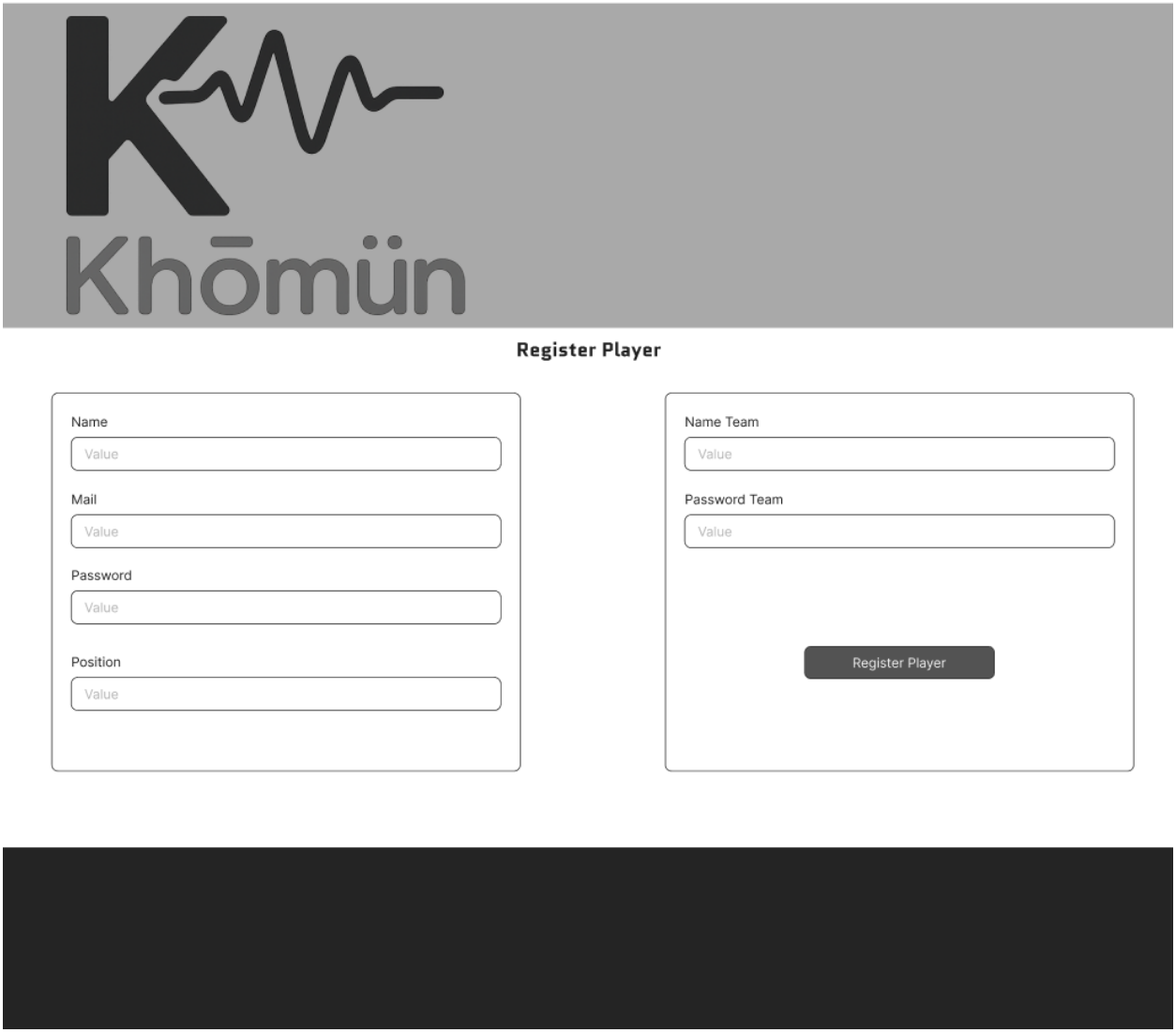


***Figura 8**- Registro Jugador*

La ***Figura 7*** presenta el **formulario de registro de jugadores en la aplicación Khomün**. La interfaz permite introducir datos básicos como nombre, correo electrónico, contraseña y posición, además de la información del equipo (nombre y clave de acceso). Este diseño facilita la creación de perfiles individuales vinculados a un equipo concreto, sentando la base para la gestión y análisis de métricas personalizadas dentro de la plataforma.

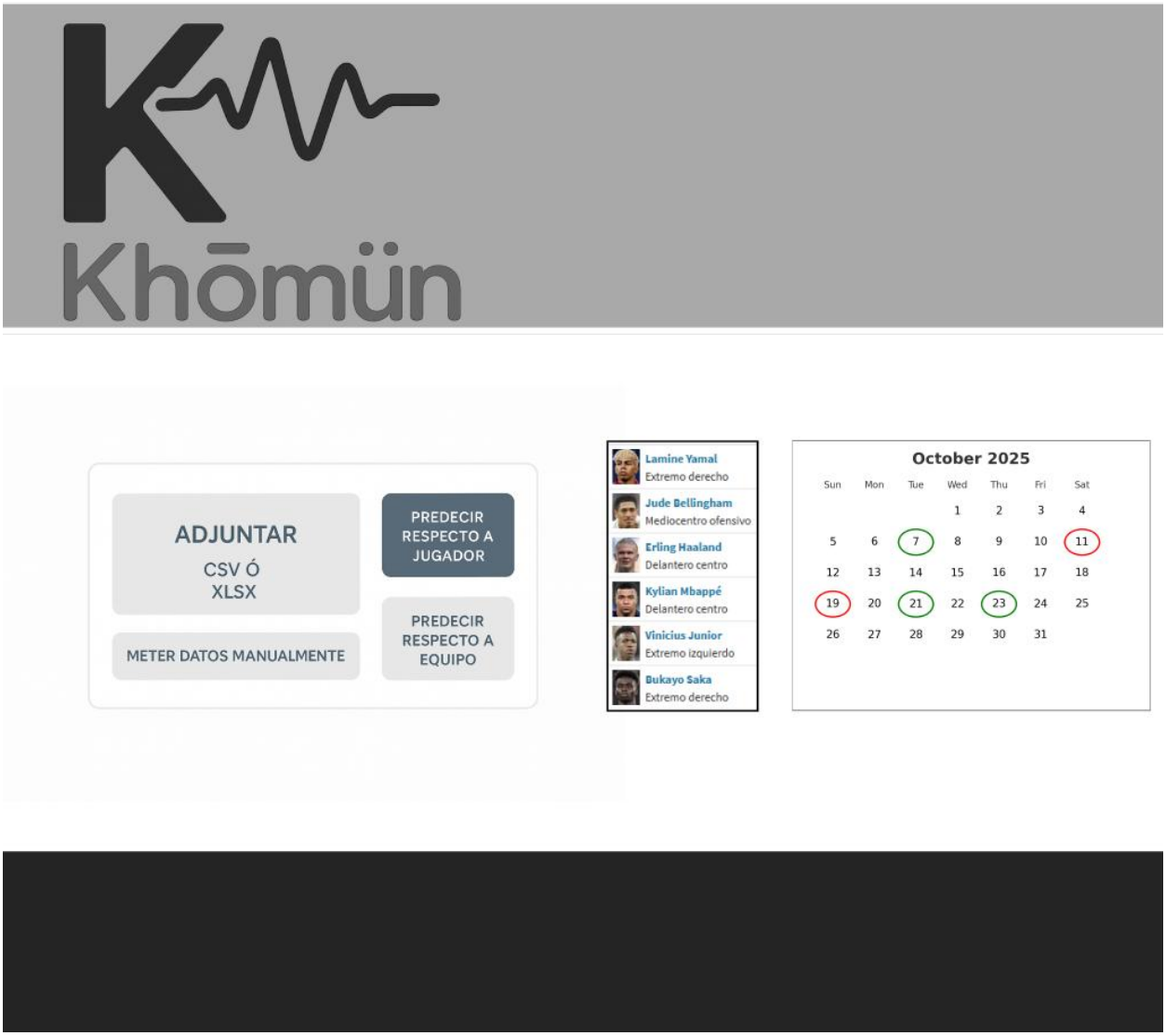


Figura 9- *Vista Entrenador*

La ***Figura 8*** muestra la **vista principal del entrenador en la aplicación Khomün**. Desde esta interfaz se pueden cargar datos en formato CSV o XLSX, introducir métricas manualmente y generar predicciones tanto a nivel individual como colectivo. Además, incluye un calendario para organizar sesiones y una lista de jugadores, lo que facilita la planificación y el análisis del rendimiento del equipo en su conjunto.

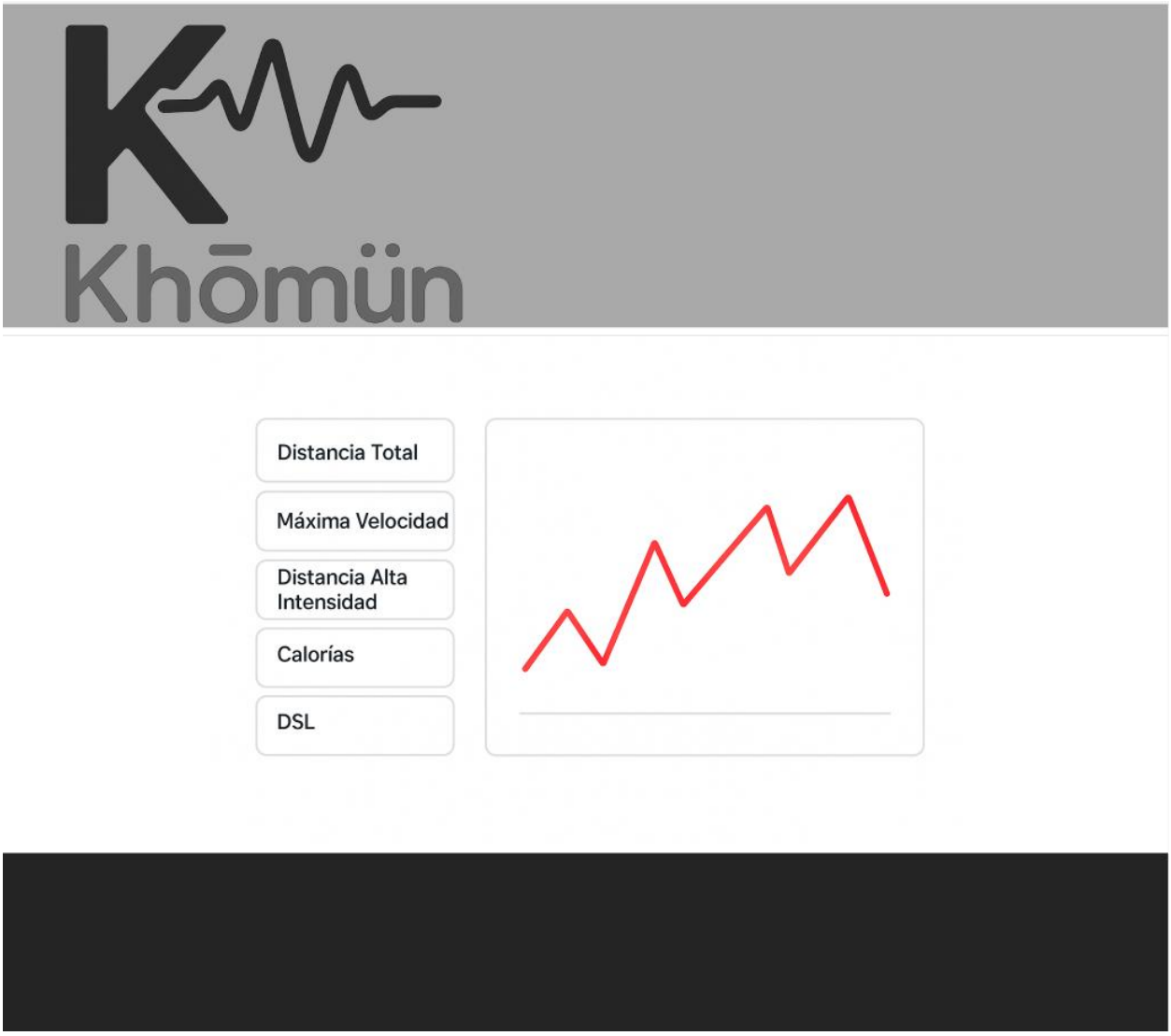


Figura 10 *-Vista Jugador*

La ***Figura 9*** representa la **vista del jugador en la aplicación Khomün**. En esta interfaz se muestran indicadores clave de rendimiento individual, como distancia total, velocidad máxima, distancia de alta intensidad, calorías y DSL. Los datos se presentan mediante gráficos que facilitan la interpretación de la evolución del jugador a lo largo de la sesión, permitiendo un seguimiento claro y visual de su carga de trabajo.

4.3 Elección de tecnologías y diseño de la arquitectura

La elección de las tecnologías no puede ser arbitraria, ya que determina en gran medida la escalabilidad, mantenibilidad y robustez del sistema. Debido a esto es necesario seleccionar un conjunto tecnológico que se ajuste a las necesidades del proyecto, así como a los estándares actuales del desarrollo de aplicaciones web y el análisis de datos.

La plataforma desarrollada en este trabajo combina la gestión de datos deportivos, la integración de modelos predictivos, la visualización de métricas en tiempo real y la capacidad de generar interfaces dinámicas y accesibles. Además, para poder integrar los modelos predictivos será necesario tener una flexibilidad en el tratamiento de datos y poder facilitar la integración de librerías de machine learning. Por ello se necesitará una arquitectura distribuida en tres capas principales: una base de datos capaz de manejar información heterogénea y con potencial de crecimiento, un backend que actúe como intermediario entre los datos, los modelos de predicción y la experiencia de usuario y un frontend que proporcione al usuario final una interfaz clara, interactiva y adaptable a diferentes contextos de uso.

Este diseño se ajusta al patrón Modelo–Vista–Controlador (MVC) adaptado a un entorno cliente–servidor moderno. En este esquema, la Vista corresponde al frontend en React, el Controlador al backend en Flask y el Modelo a la base de datos en MongoDB junto con los servicios de *machine learning*. Así, la Vista solicita información, el Controlador gestiona la petición y el Modelo devuelve los datos procesados, garantizando una arquitectura modular y escalable.

4.3.1 Base de datos: MongoDB

Para el almacenamiento de la información se ha optado por MongoDB, una base de datos NoSQL orientada a documentos. La elección se fundamenta en la naturaleza de los datos utilizados en la plataforma, caracterizados por ser heterogéneos y, en muchos casos, semiestructurados teniendo algunas variaciones entre colecciones. Para empezar, se crearán colecciones para los **usuarios**, **equipos** y **sesiones**, estando las tres relacionadas mediante unos identificadores específicos de cada colección.

Estas colecciones contarán con los siguientes campos:

- **Usuarios:** user_id, de tipo Integer, user_name, de tipo String, user_email, de tipo String, de tipo user_type, de tipo Integer, de tipo user_position (en el caso de los jugadores), de tipo String y password, de tipo String hasheado.
- **Equipos:** team_id, de tipo Integer, team_name, de tipo String, dt_id, de tipo Integer, sport_type, de tipo String, password, de tipo String y players, de tipo Array de Integers.
- **Sesiones:** session_id, de tipo Integer, player_id, de tipo Integer, team_id, de tipo Integer, sport_type, de tipo Integer, session_type, de tipo Integer, session_date, de tipo String, total_distance, de tipo Double, impacts, de tipo Integer, max_vel, de tipo Double, high_intensity_distance, de tipo Double, calories, de tipo Integer, dsl de tipo Integer.

4.3.2 Backend: Python con Flask

En cuanto a la capa de servidor, se ha seleccionado Python junto con el microframework Flask. El uso de Flask se justifica por su ligereza, modularidad y simplicidad de configuración, lo que permite construir de manera ágil una API RESTful encargada de gestionar la comunicación entre la base de datos y la interfaz de usuario. Además, su integración con librerías de procesado de datos y modelos predictivos resulta directa, facilitando la implementación de la lógica necesaria para calcular y servir las métricas deportivas en tiempo real.

4.3.3 Frontend: React

Para la capa de presentación se ha optado por React, una de las bibliotecas más extendidas para el desarrollo de interfaces web modernas. Su elección se basa en su capacidad para construir interfaces dinámicas e interactivas, aspecto fundamental en un sistema que requiere mostrar métricas y gráficas de evolución de manera clara y accesible.

React se estructura en torno a componentes reutilizables, lo que favorece la escalabilidad, el mantenimiento del código y la adaptación a futuras ampliaciones del

sistema. Finalmente, su integración con APIs RESTful resulta especialmente fluida, lo que garantiza una comunicación eficiente con el backend desarrollado en Flask.

En conjunto, la combinación de MongoDB, Flask y React constituye un stack tecnológico que proporciona flexibilidad, escalabilidad y robustez y una muy buena conexión entre las distintas tecnologías gracias a que tanto React como Flask utilizan JSON para comunicarse y MongoDB utiliza colecciones en BSON. Al mismo tiempo que se ajusta a las necesidades específicas de esta plataforma (destinada al análisis y predicción de métricas deportivas en contextos de entrenamientos y partidos de fútbol 11 y fútbol sala).

4.4 Configuración del entorno

La correcta configuración del entorno de desarrollo constituye un aspecto fundamental para garantizar la reproducibilidad y la coherencia del sistema implementado que permite la integración fluida de todos los componentes de la arquitectura descrita en el apartado anterior.

El desarrollo se llevó a cabo sobre un entorno local, en un equipo con sistema operativo con Windows 10 y características de hardware suficientes para el procesamiento de datos como un Ryzen 3 1600 y 8 gigas de RAM, sin necesidad de utilizar una GPU para el entrenamiento de modelos.

Gracias a que estas características no son especialmente altas el desarrollo de la plataforma tiene una fácil replicabilidad y escalabilidad en otros entornos. Para la base de datos se utilizó la herramienta gráfica MongoDB Compass, que facilitó tanto la gestión de colecciones como la verificación de los datos almacenados. Para la edición de código se empleó el entorno de desarrollo Visual Studio Code, que permitió gestionar tanto el frontend como el backend de forma centralizada, con soporte de extensiones útiles para la depuración y el formateo de código.

La base de datos necesitará tener instalado MongoDB Compass. Luego para inicializarla en el puerto 27017 será necesario abrir una terminal y ejecutar el comando:

- `mongod`

El backend necesitará tener instalado Python y un requirements.txt para poder instalar las librerías.

- `pip install -r requirements.txt`

Con esto para levantar el servidor en el puerto 5020 se tendrá que utilizar una terminal para ir a la carpeta donde se encuentre el wsgi y ejecutar el siguiente comando:

- `python .\wsgi.py`

El frontend necesitará tener instalado Node.js y npm y para levantar react con los módulos que se utilizarán es necesario ejecutar los siguientes comandos en una terminal:

- `npx create-react-app [fileName]`
- `npm install react-router-dom axios sweetalert2 react-chartjs-2 chart.js react-calendar`

Para levantar la aplicación en el puerto 3000 respectivamente, se abrirá una terminal y se ejecutará el siguiente comando:

- `npm start`

El código tanto del backend como del frontend están almacenados en la nube en dos repositorios de git.

Gracias a este punto y el resto de la configuración descrita con anterioridad asegura la replicabilidad del sistema y proporciona una base sólida para su despliegue en entornos de producción o su ampliación con nuevas funcionalidades.

4.5 Pruebas de la aplicación

- **Registro de Usuarios**:

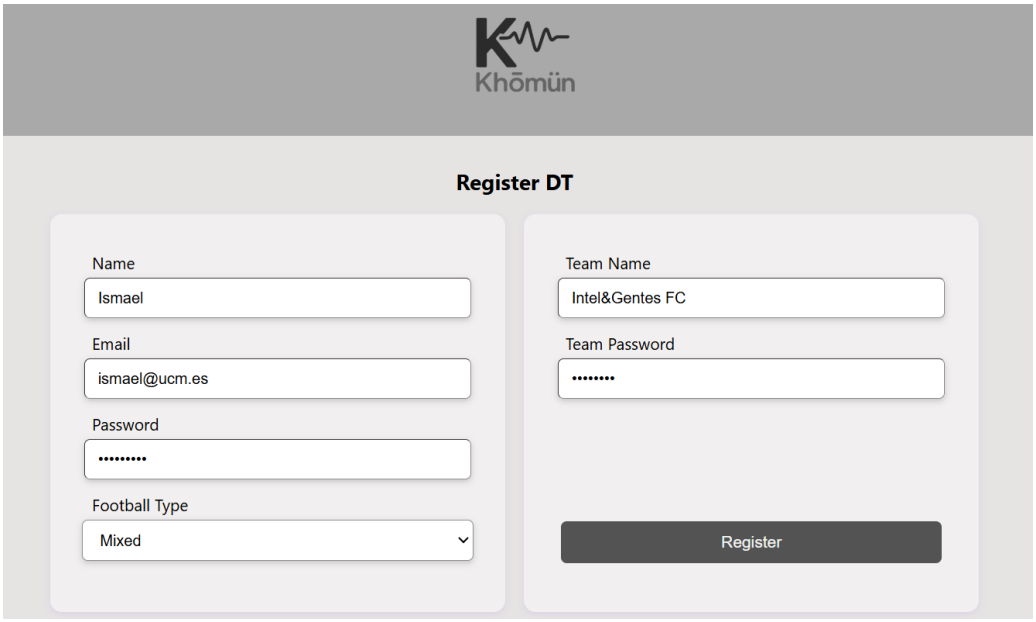


***Figura 11**- Registro DT*

En el registro de usuarios tenemos dos versiones bastante similares que consisten en dos formularios con pequeñas variaciones entre los distintos tipos de usuario. En la ***Figura 11*** se registra el entrenador, introduciendo sus datos junto con los del nuevo equipo y en la ***Figura 12*** se registra un jugador con sus datos y los datos del equipo ya creado.

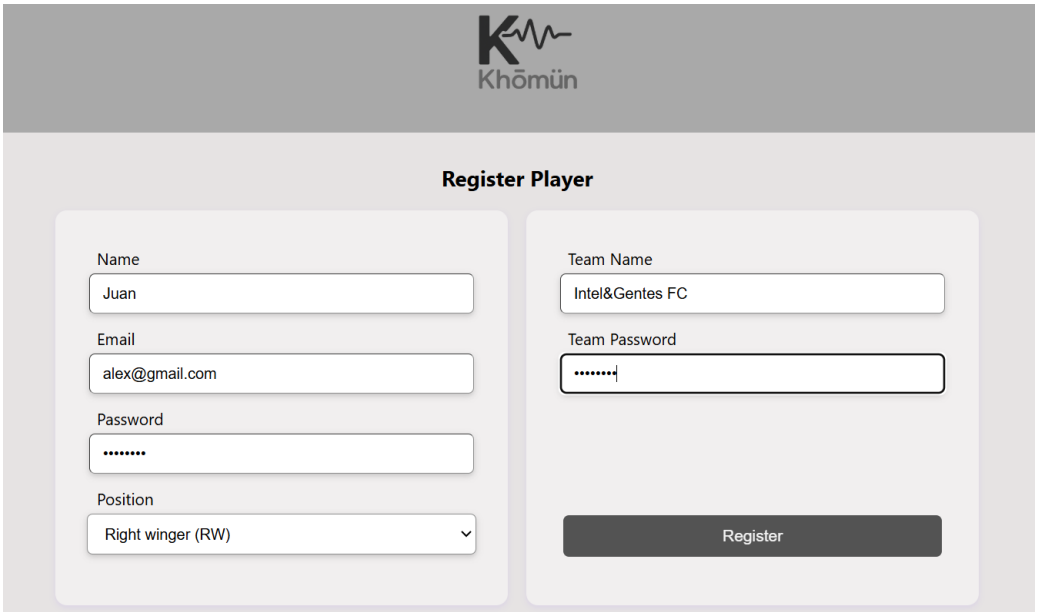


***Figura 12**-Registro Jugador*

- **Inicio de Sesión**:

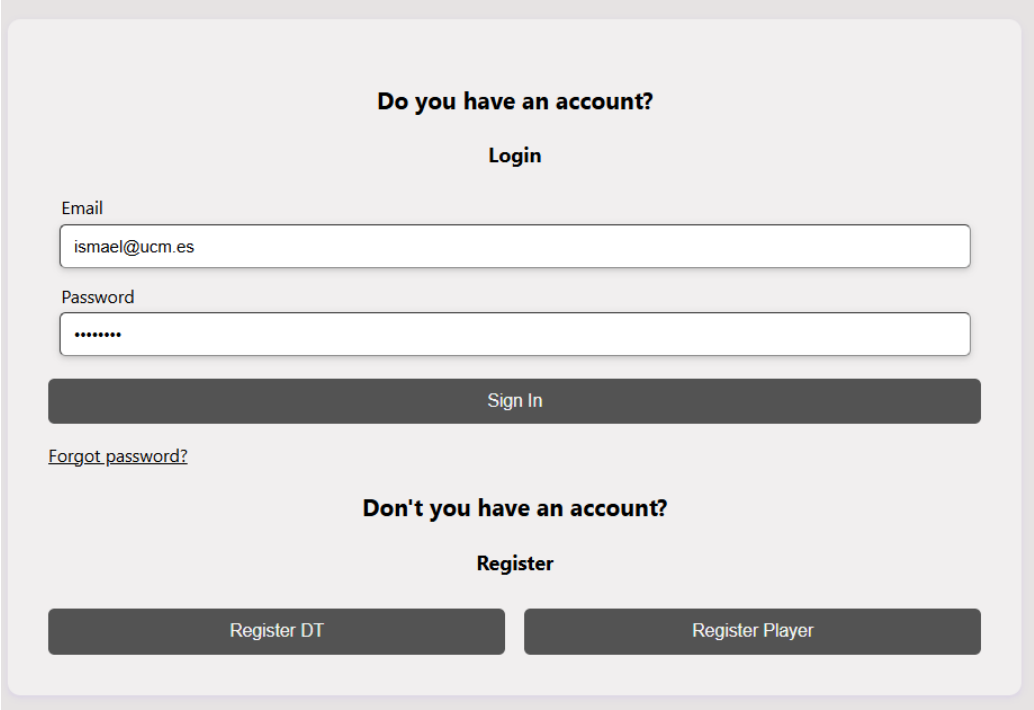


***Figura 13**-Inicio de sesión*

En el inicio de sesión se cuenta con un formulario de autenticación que permite a los usuarios acceder de manera segura mediante correo electrónico y contraseña.

- **Predicciones**:

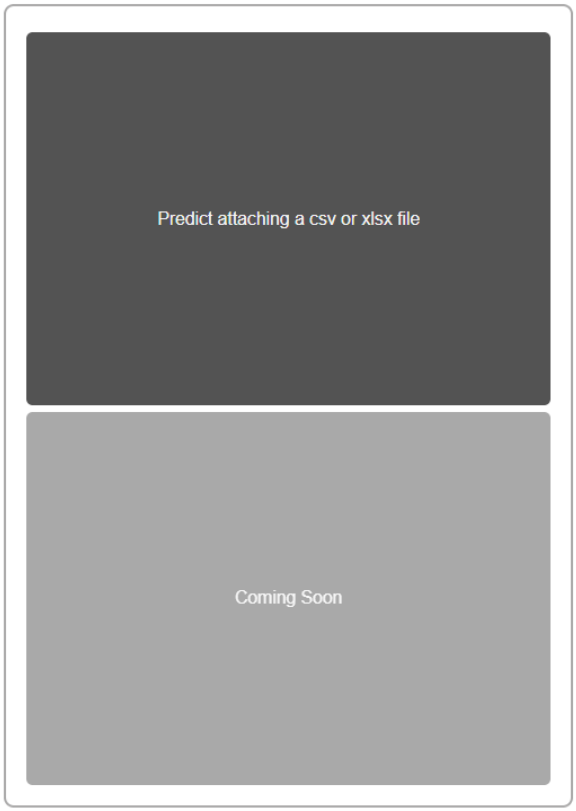


***Figura 14**- Selección de base con la que predecir*

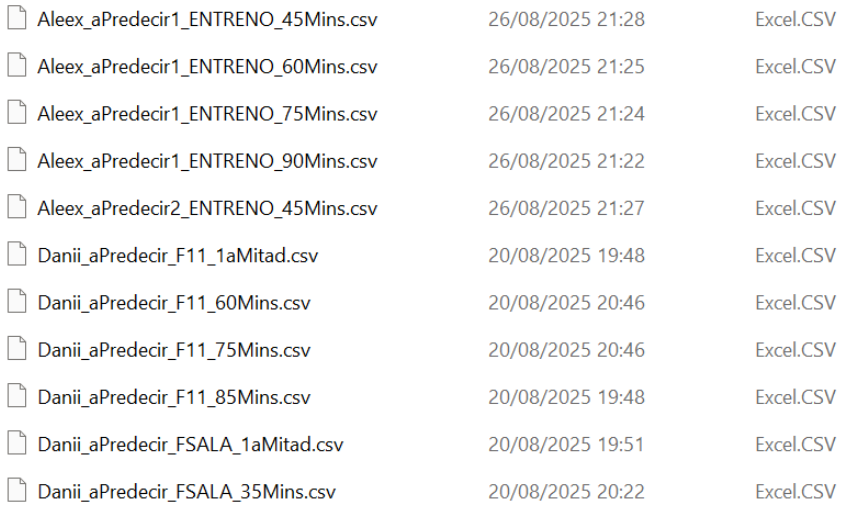


***Figura 15**- Selección del CSV a predecir*

***Figura 16**-Selección de modelo de Machine Learning y jugador*

***Figura 17** – Resultado de la predicción*

El proceso para realizar una predicción consiste en seleccionar un archivo CSV o XLSX con los datos de una sesión. A continuación, el entrenador elige un modelo de predicción con el que trabajar, así como el jugador al que corresponden los datos cargados. Por último, se ofrecen dos opciones, si realizar la predicción utilizando un modelo entrenado específicamente con los registros de ese jugador o, alternativamente, empleando un modelo general entrenado con la información de todo el equipo.

Una vez completados estos pasos, el sistema devuelve los resultados de la predicción, mostrando el valor de DSL estimado, el porcentaje de fiabilidad del modelo y una interpretación sobre el estado físico esperado del jugador al final de la sesión. Como referencia, en las pruebas realizadas, utilizando los ficheros CSV que se muestran en la ***Figura 17*** la fiabilidad de los modelos se encontraría entre un 79% y un 93%.

- **Visualización de métricas**:

***Figura 18**-Vista jugadores del equipo*

El entrenador dispondrá de una lista con sus jugadores en la pantalla principal donde podrá buscar un jugador en concreto y con un simple clic en su "carta de jugador" observar sus métricas como se ve en la ***Figura 18***.

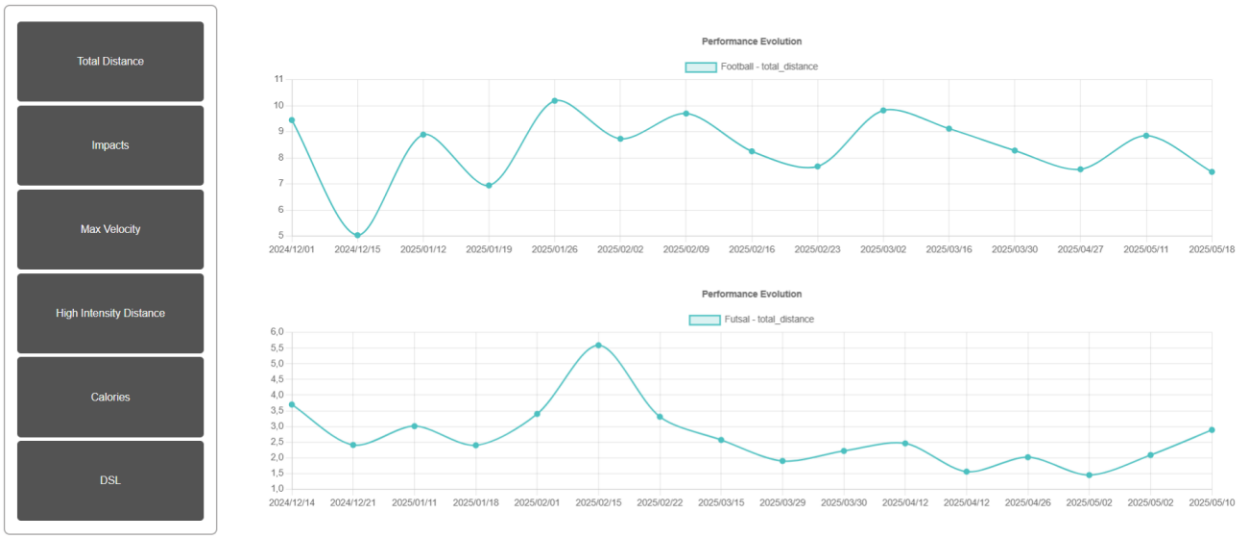


***Figura 19**-Métricas jugador mixto*

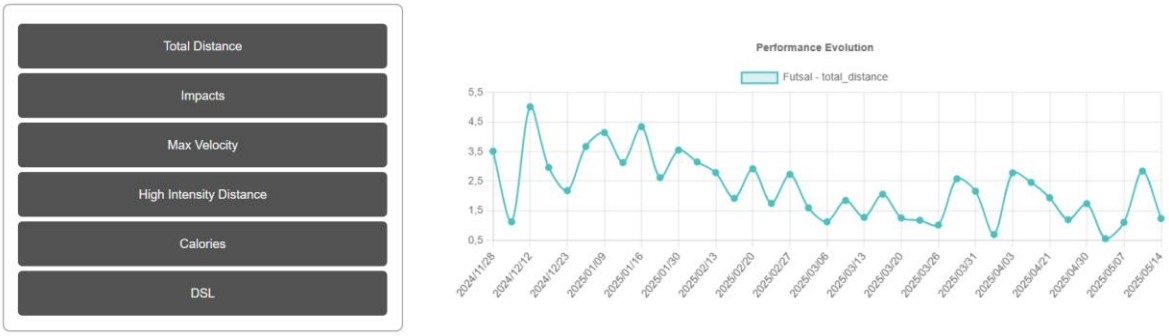


***Figura 20**-Métricas jugador*

Las métricas tienen dos modos de visualizarse dependiendo de si se práctica más de un deporte y si existen entrenamientos y partidos de un mismo jugador o si solo se practica un deporte y solamente se tienen datos de partidos o de entrenamientos. La ***Figura 19*** muestra la distancia total de Francisco Daniel Ortiz en partidos de fútbol 11 y fútbol sala y la ***Figura 20*** muestra la distancia total de Alejandro Nafría Medina en entrenamientos de fútbol sala.

- **Calendario**:

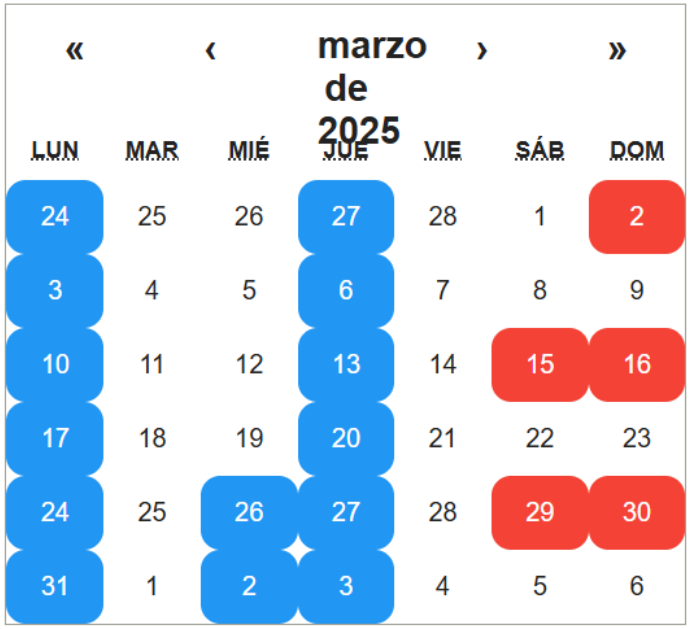


***Figura 21**-Calendario sesiones*

El calendario; ***Figura 21***, sirve para poder visualizar cuando se realizaron entrenamientos y partidos. Los coloreados en azul son los entrenamientos y los coloreados en rojo son los partidos, aquí sin distinción de tipo.

4.6 Errores encontrados

Durante el desarrollo de la plataforma se han encontrado ciertas limitaciones que conviene destacar, tanto por su impacto en la versión actual como por su relevancia a la hora de ampliar la plataforma en un futuro hacia algo más comercial.

- **Equipos mixtos**: Los registros procedían de fuentes heterogéneas: entrenamientos realizados por Alejandro Nafría y partidos de fútbol sala y fútbol 11 realizados por Francisco Daniel Ortiz. Con el fin de agrupar todos los datos en un mismo contexto y mantener una consistencia entre los datos y los jugadores además de los equipos se optó por crear artificialmente la categoría de deporte "mixto", haciendo alusión a la unión en un único equipo de fútbol 11 y fútbol sala. Esta opción permitió avanzar en el desarrollo, pero no presenta una opción realista en un entorno profesional.
    - **Posible solución:** la opción de equipos mixtos se eliminará cuando se disponga de un volumen suficiente de datos consistentes y diferenciados por modalidad, en este caso fútbol 11 y fútbol sala ya que los entrenamientos y los partidos en conjunto quedarán como la norma. Así cada jugador quedará vinculado a un único equipo y a una única modalidad de deporte.
- **Predicción basada en archivos CSV/XLSX:** tanto los conjuntos de datos utilizados para el entrenamiento como los empleados en la inferencia han de tener una estructura idéntica, limitando la flexibilidad y dificultando la incorporación de nuevos registros y complicando enormemente la agilidad a la hora obtener una predicción en momentos donde la velocidad del proceso apremia, como podría ser el descanso de un partido.
    - **Posible solución:** para superar la rigidez actual, por una parte, se plantea que el entrenamiento de los modelos se realice de manera nativa sobre las colecciones de MongoDB, evitando la dependencia de archivos

CSV/XLSX externos. De este modo el modelo se adaptará de forma dinámica a la información almacenada correspondiente a un jugador o al equipo en su totalidad. Asimismo, se propone integrar la API de STATSports con el objetivo de obtener los datos en tiempo real, aplicar un preprocesado inmediato y generar las predicciones de forma automática, logrando así un proceso más ágil, natural y eficiente para el usuario.

- **Estilo visual:** si bien las funcionalidades principales descritas en el apartado 3.1 son plenamente operativas desde la perspectiva funcional, las limitaciones en la personalización del estilo provocaron que el resultado final fuese poco atractivo desde el punto de vista estético. El componente más afectado es el calendario donde se consiguió mostrar la información asociada pero no se logró dotarlo de una apariencia clara y homogénea.
    - **Posible solución:** en relación con el calendario, se plantea la sustitución o extensión del componente actual por alternativas más personalizables que permitan un diseño visual atractivo y coherente con el resto de la interfaz.

Capítulo 5 - Conclusiones y trabajo futuro

A continuación, se presentan las conclusiones de nuestro proyecto, evaluando tanto el planteamiento inicial como los resultados obtenidos, y proponiendo mejoras e ideas para futuras versiones de la aplicación.

El objetivo de este TFG fue combinar dos de nuestros grandes pilares —la formación universitaria y el deporte— tomando como punto de partida el trabajo "Generación de una herramienta de Machine Learning para apoyar a deportistas" de nuestros compañeros Juan Israel Baroffi González y Javier Parra González en 2024 y adaptándolo al ámbito futbolístico.

Desde el inicio, nos enfrentamos a los retos habituales en proyectos de machine learning. En primer lugar, debimos filtrar y seleccionar las variables más relevantes de entre el amplio volumen de datos bruto disponible en nuestro "cachivache". Gracias al desarrollo del coeficiente de correlación de Pearson y matrices de correlación, conseguimos estructurar la información y reducirla a un conjunto fijo de columnas. Tras varios meses dedicados a la recopilación, limpieza y organización de estos datos en nuestro fichero CSV, procedimos a entrenar y evaluar distintos algoritmos de aprendizaje automático, alcanzando resultados muy satisfactorios.

Por otro lado, nos establecimos como objetivo secundario, la publicación del dataset de entrenamiento, que hemos realizado en la plataforma Kaggle y puede consultarse en la siguiente URL:

https://www.kaggle.com/datasets/alejandronafria/football-fatigue-metrics-dsl

Como próximas líneas de trabajo, identificamos varias limitaciones clave que quisiéramos superar:

1. **Temporalidad de la muestra**: un periodo de recolección más prolongado habría facilitado el uso de algoritmos alternativos y la comparación de su precisión.
2. **Ampliar la muestra de evaluación**: en una fase inicial intentamos incluir como sujeto de prueba a Mario, un compañero de Alejandro del segundo equipo, pero

la logística de prestarle el chaleco sensorizado (coordinación de entrenamientos, sincronización de dispositivos y limpieza de esos datos) resultó demasiado compleja.

Para futuras iteraciones planificamos:

2.1 Incorporar chalecos adicionales o acuerdos de préstamo para facilitar el acceso simultáneo a varios jugadores.

2.2 Establecer un protocolo de recogida de datos estandarizado (horarios, formato de exportación, filtros automáticos) que minimice el tiempo de postprocesado y garantice la calidad de la información.

2.3 Adquisición de una cinta de frecuencia cardíaca: permitirá medir ritmo y variabilidad cardíaca con mayor precisión durante todo el entrenamiento y el partido. Al integrarlo con nuestro sistema, conseguiremos curvas de carga más detalladas, mejoraremos la exactitud de los algoritmos predictivos y enriqueceremos los informes de rendimiento para el cuerpo técnico.

3. **Mejoras tecnológicas y funcionales para el sistema:**

- Desarrollo de una **aplicación móvil** que facilite tanto la visualización de métricas en tiempo real como la gestión de datos por parte de los entrenadores.
- Obtener la **API oficial de STATSports**, lo que nos permitiría acceder a métricas más precisas y completas que las obtenidas mediante exportaciones manuales.
- Implementar la opción de **añadir datos de forma manual** en la aplicación, útil para registrar incidencias, observaciones o métricas no capturadas por el chaleco.
- Entrenar **modelos de machine learning más complejos** (redes neuronales profundas, modelos secuenciales, etc.) para mejorar la robustez y capacidad predictiva.
- Permitir que el sistema **trabaje directamente con la base de datos**, en lugar de depender de archivos añadidos manualmente, lo que aumentaría el realismo y la escalabilidad.

- Incorporar la opción de **guardar entrenamientos y partidos directamente en la base de datos**, creando un historial estructurado de sesiones.
- Desarrollar la posibilidad de **visualizar gráficas por sesión**, tanto de un único jugador como de un grupo completo, favoreciendo la comparación de perfiles.
- Ampliar las capacidades de predicción para estimar métricas en diferentes horizontes temporales (**60, 75 minutos, etc.**) y no solo al final del partido o entrenamiento.
- Extender las predicciones a nuevas dimensiones, como **carga de trabajo acumulada o fatiga extra**, aportando más valor al cuerpo técnico.
- Mejorar el **apartado visual** de la plataforma, por ejemplo, incorporando rangos de colores que resalten indicadores críticos como el DSL tras la predicción, facilitando una lectura rápida e intuitiva.
- Crear una **herramienta automatizada** para la generación de CSVs en el formato adecuado a partir de los datos obtenidos desde la API, de forma que se reduzca el trabajo manual de preparación de datos y se agilice el flujo de entrada al sistema.
- Proponer como línea de trabajo futuro el despliegue de la plataforma mediante contenedores (**Docker**), lo que permitiría una mayor portabilidad, escalabilidad y facilidad de mantenimiento en entornos de producción.

Estas iniciativas consolidarán la capacidad del proyecto para ofrecer predicciones más fiables y un valor añadido real al proceso de toma de decisiones deportivas.

A pesar de estas restricciones, estamos muy satisfechos con el prototipo desarrollado y confiamos en que las propuestas aquí descritas servirán de impulso para futuras mejoras. Los resultados de las pruebas demuestran que el modelo proporciona predicciones con un nivel de precisión satisfactorio. Las imágenes de estos estudios se muestran en el Apéndice-A

A continuación, mostramos una lista relacionando diferentes asignaturas de nuestros grados y los conocimientos adquiridos que han facilitado este TFG:

Base de datos

Aprendimos a analizar y crear instrucciones SQL para la manipulación, definición y control de esquemas y datos en un Sistema de Gestión de Bases de Datos Relacional, además de garantizar la integridad referencial y la coherencia de la información.

Base de datos NoSQL

Nos enfocamos en evaluar, proponer y presentar soluciones eficientes de almacenamiento, discerniendo cuándo es más apropiado emplear un modelo relacional o NoSQL, y estudiamos las características y casos de uso de las familias documental, clave-valor, columnar y de grafos.

Sistemas Web (GIC) Aplicaciones Web (GII)

Adquirimos la habilidad de argumentar decisiones de diseño en aplicaciones web complejas, combinar patrones en cliente y servidor para integrar servicios de apoyo, construir documentos HTML5 correctos, desarrollar la persistencia de datos en bases de datos y diseñar contramedidas ante riesgos de seguridad.

Sistemas Inteligentes (GIC)

Consolidamos el conocimiento de algoritmos de búsqueda (informada, no informada, local y con adversario), el diseño de sistemas basados en agentes, la elección de representaciones adecuadas, la modelización del conocimiento y la gestión de la incertidumbre, así como la interoperabilidad entre agentes y la aplicación de técnicas (algoritmos genéticos, redes neuronales, sistemas de reglas) para dotar de inteligencia a sistemas complejos.

Fundamentos de la Programación I

Adquirimos los conocimientos básicos para el uso y la programación de ordenadores, sistemas operativos, bases de datos y aplicaciones informáticas con vocación ingenieril. Desarrollamos la capacidad de analizar y sintetizar información para resolver problemas, integrar creativamente distintos conocimientos en la resolución de retos informáticos aplicando el método científico y tomar decisiones de diseño al elaborar prácticas y ejercicios.

Además, reforzamos habilidades transversales como la comunicación técnica, el juicio crítico en la interpretación de datos relevantes y la autonomía necesaria para emprender estudios posteriores.

Fundamentos de la Programación II

Profundizamos en el desarrollo y validación de programas en lenguajes de programación concretos, evaluando la eficiencia de distintos algoritmos para seleccionar el más adecuado y manejando estructuras de datos persistentes a través de ficheros. Desarrollamos la capacidad de analizar un problema, diseñar la solución y validar su implementación mediante ejercicios de programación, y adquirimos destrezas en el uso de herramientas informáticas y sistemas operativos para ejecutar y depurar nuestros programas. Asimismo, reforzamos la autonomía investigadora y el juicio crítico al reunir e interpretar datos relevantes, y ejercitamos la comunicación de ideas y soluciones tanto a públicos especializados como no especializados.

Tecnología de la Programación I

En esta asignatura fortalecimos los conocimientos básicos de uso y programación de ordenadores, sistemas operativos y bases de datos con aplicación ingenieril, y adquirimos la capacidad de analizar, diseñar, construir y mantener aplicaciones robustas, seguras y eficientes, eligiendo el paradigma y los lenguajes más adecuados. Desarrollamos habilidades de comunicación oral y escrita —en inglés y español— empleando medios audiovisuales y trabajando en equipo multidisciplinar, así como competencias de análisis y síntesis para la resolución de problemas informáticos mediante el método científico. Aprendimos a argumentar y tomar decisiones de diseño en cada práctica, a programar y depurar aplicaciones orientadas a objetos en lenguajes concretos utilizando entornos de desarrollo integrados y a validar nuestros desarrollos con criterios de calidad.

Gestión de la Información en la Web

En esta asignatura adquirimos el conocimiento y la aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a sistemas de información, incluidos los basados en la web. Desarrollamos la capacidad de concebir sistemas, aplicaciones y servicios sobre tecnologías de red —Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil—,

al mismo tiempo que aprendimos a garantizar su seguridad e integridad. Además, ejercitamos nuestras habilidades de comunicación oral y escrita en inglés y español empleando medios audiovisuales y la gestión creativa de información mediante el método científico. Como resultados de aprendizaje, somos capaces de implementar mecanismos de explotación de datos accesibles en la web, crear aplicaciones web seguras para gestionar información y colaborar eficazmente en equipos de desarrollo de software

CONTRIBUCIONES PERSONALES

Es importante señalar que ambos integrantes han contribuido de manera equitativa en la elaboración de este Trabajo de Fin de Grado.

Francisco Daniel Ortiz Delgado

Contribuciones del estudiante:

Recopilación inicial de ideas para el desarrollo del trabajo

Desde las primeras fases del proyecto se llevó a cabo una lluvia de ideas con el fin de definir los objetivos y el alcance del trabajo. En este proceso se exploraron diferentes enfoques metodológicos y tecnológicos hasta concretar una propuesta sólida. Como parte de esta fase, se contactó con la empresa **Beyond Stats** con el objetivo de recibir orientación y validar la viabilidad de utilizar tecnología GPS aplicada al análisis del rendimiento deportivo sin ningún éxito. Una vez enfocada la idea del trabajo toco decidir que dispositivo era el más idóneo para el desarrollo del estudio.

Comparativa de los mejores chalecos de fútbol con GPS

	CHALECO CATAPULT ONE GPS	CHALECO STATSPORTS APEX	GPS DE FÚTBOL OLIVER
			
	VER PRODUCTO	VER PRODUCTO	VER PRODUCTO
APP GRATUITA	✔	✔	✔
VENTA PACK COMPLETO	✔	✔	✔
OPINIONES	★★★★★	★★★★★	★★★★
DISPONIBILIDAD PRIME	✔	✔	✔
VALORACIÓN LIVING FOOTBALL	8,5/10	9,5/10	7/10
	VER PRODUCTO	VER PRODUCTO	VER PRODUCTO

Figura 22 *-Comparación dispositivos*

Una vez sopesada las opciones posibles gracias a la ***Figura 22*** que muestra una comparativa [10] entre las posibles opciones nos decantamos por StatSports.

Formación inicial en machine learning

A diferencia de mi compañero, que ya contaba con una base sólida en machine learning adquirida en la asignatura de Sistemas Inteligentes, mi punto de partida en esta materia fue más limitado. Esto supuso un reto adicional al inicio del proyecto, ya que tras la fase de lluvia de ideas resultaba imprescindible contar con unos conocimientos mínimos para poder avanzar de manera conjunta y equilibrada en las siguientes etapas.

Por ello, me vi en la necesidad de realizar un proceso de aprendizaje intensivo en un corto periodo de tiempo. Durante estas primeras semanas dediqué un esfuerzo significativo a comprender los fundamentos teóricos y prácticos del aprendizaje automático, apoyándome en material académico proporcionado por Ismael. Gracias a este trabajo de autoformación conseguí ponerme al día y alcanzar un nivel que me permitió seguir el ritmo de mi compañero y participar activamente en las fases de análisis de datos, experimentación y validación de modelos.

Resolución de problemas iniciales con el chaleco y feedback con STATSports

Una de las primeras dificultades encontradas fue la configuración y correcto funcionamiento del chaleco GPS adquirido a STATSports. Para resolver estos problemas se realizaron diversas pruebas técnicas y se mantuvo comunicación directa con el servicio de soporte de la empresa, aportando feedback sobre errores detectados y soluciones aplicadas. Esta labor fue esencial para garantizar la correcta recolección de datos en fases posteriores.

Publicación del CSV en Kaggle

Con el fin de contribuir a la comunidad científica y facilitar la reproducibilidad del trabajo, se decidió publicar el dataset generado en la plataforma **Kaggle**. Este proceso implicó la preparación del archivo CSV, su anonimización y la inclusión de una descripción detallada de las variables, asegurando que otros investigadores pudieran reutilizar los datos de forma transparente.

Recopilación de datos siendo nosotros los deportistas

Para nutrir el dataset, los propios integrantes del proyecto actuamos como deportistas, utilizando el chaleco GPS en entrenamientos y partidos. Esta contribución no solo permitió disponer de un conjunto de datos real y específico, sino también comprender de primera mano las limitaciones y características del dispositivo en contextos prácticos.

Búsqueda de estudios y trabajos relacionados

Se llevó a cabo una revisión bibliográfica exhaustiva con el objetivo de contextualizar el proyecto en el marco de la investigación actual. Para ello se consultaron artículos científicos, tesis y reportes técnicos relacionados con **machine learning en el deporte**, la monitorización mediante GPS y las métricas de rendimiento en fútbol y fútbol sala. Este trabajo documental resultó clave para fundamentar la metodología propuesta.

Limpieza y preparación de datos

Antes de aplicar los modelos predictivos fue necesario realizar un exhaustivo proceso de limpieza y preparación de datos. Este incluyó la detección y gestión de valores nulos, la normalización de variables, la eliminación de redundancias y la organización del dataset en un formato adecuado para su posterior análisis. Esta fase fue determinante para garantizar la calidad de los resultados.

Estudio de códigos en Python de machine learning

Una parte importante de la formación personal durante el TFG consistió en el estudio y análisis de diversos scripts en **Python** aplicados al machine learning. Se experimentó con diferentes modelos predictivos, evaluando su rendimiento y adaptándolos a las características del dataset. Esto supuso un aprendizaje progresivo en bibliotecas como **scikit-learn, XGBoost y pandas**, entre otras.

Desarrollo de la aplicación:

Figma

Para la fase de diseño se empleó Figma, lo que permitió crear prototipos de la aplicación, validar la usabilidad de la interfaz y definir el estilo visual del proyecto antes de proceder al desarrollo técnico.

MongoDB

Se diseñó e implementó una base de datos en MongoDB para almacenar y gestionar la información recopilada. La elección de esta tecnología permitió un manejo flexible de datos semiestructurados y facilitó la escalabilidad del proyecto.

Flask

Se empleó Flask para el desarrollo del backend, creando una API ligera y modular en Python. Su uso permitió definir endpoints para la gestión de usuarios y la conexión con los modelos de predicción, además de integrar de forma eficiente la base de datos en MongoDB y la interfaz en React.

React

En la parte del frontend se desarrolló una interfaz con React, orientada a la visualización de métricas y resultados de los modelos. Este trabajo implicó el diseño de componentes reutilizables, la integración con el backend y la optimización de la experiencia de usuario.

Memoria

Finalmente, se elaboró la **memoria del TFG**, donde se recogió todo el proceso seguido: desde la motivación inicial y el marco teórico, hasta la metodología aplicada, los resultados obtenidos y las conclusiones. Esta redacción requirió un esfuerzo de síntesis, rigor académico y una adecuada estructuración para reflejar de manera clara y coherente el trabajo realizado.

Alejandro Nafría Medina

Contribuciones del estudiante:

Recopilación inicial de ideas para el desarrollo del trabajo

Desde las primeras fases del proyecto se llevó a cabo una lluvia de ideas con el fin de definir los objetivos y el alcance del trabajo. En este proceso se exploraron diferentes enfoques metodológicos y tecnológicos hasta concretar una propuesta sólida. Como parte de esta fase, se contactó con la empresa **Beyond Stats** con el objetivo de recibir orientación y validar la viabilidad de utilizar tecnología GPS aplicada al análisis del rendimiento deportivo sin ningún éxito. Una vez enfocada la idea del trabajo toco decidir que dispositivo era el más idóneo para el desarrollo del estudio.

Comparativa de los mejores chalecos de fútbol con GPS

	CHALECO CATAPULT ONE GPS	CHALECO STATSPORTS APEX	GPS DE FÚTBOL OLIVER
			
	VER PRODUCTO	VER PRODUCTO	VER PRODUCTO
APP GRATUITA	✔	✔	✔
VENTA PACK COMPLETO	✔	✔	✔
OPINIONES	★★★★★	★★★★★	★★★★
DISPONIBILIDAD PRIME	✔	✔	✔
VALORACIÓN LIVING FOOTBALL	8,5/10	9,5/10	7/10
	VER PRODUCTO	VER PRODUCTO	VER PRODUCTO

***Figura 23**-Comparación dispositivos*

Una vez sopesada las opciones posibles gracias a la ***Figura 23*** que muestra una comparativa [9] entre las posibles opciones nos decantamos por StatSports.

Resolución de problemas iniciales con el chaleco y feedback con STATSports

Una de las primeras dificultades encontradas fue la configuración y correcto funcionamiento del chaleco GPS adquirido a STATSports. Para resolver estos problemas se realizaron diversas pruebas técnicas y se mantuvo comunicación directa con el servicio de soporte de la empresa, aportando feedback sobre errores detectados y soluciones aplicadas. Esta labor fue esencial para garantizar la correcta recolección de datos en fases posteriores.

Publicación del CSV en Kaggle

Con el fin de contribuir a la comunidad científica y facilitar la reproducibilidad del trabajo, se decidió publicar el dataset generado en la plataforma **Kaggle**. Este proceso implicó la preparación del archivo CSV, su anonimización y la inclusión de una descripción detallada de las variables, asegurando que otros investigadores pudieran reutilizar los datos de forma transparente.

Recopilación de datos siendo nosotros los deportistas

Para nutrir el dataset, los propios integrantes del proyecto actuamos como deportistas, utilizando el chaleco GPS en entrenamientos y partidos. Esta contribución no solo permitió disponer de un conjunto de datos real y específico, sino también comprender de primera mano las limitaciones y características del dispositivo en contextos prácticos.

Búsqueda de estudios y trabajos relacionados

Se llevó a cabo una revisión bibliográfica exhaustiva con el objetivo de contextualizar el proyecto en el marco de la investigación actual. Para ello se consultaron artículos científicos, tesis y reportes técnicos relacionados con **machine learning en el deporte**, la monitorización mediante GPS y las métricas de rendimiento en fútbol y fútbol sala. Este trabajo documental resultó clave para fundamentar la metodología propuesta.

Limpieza y preparación de datos

Antes de aplicar los modelos predictivos fue necesario realizar un exhaustivo proceso de limpieza y preparación de datos. Este incluyó la detección y gestión de

valores nulos, la normalización de variables, la eliminación de redundancias y la organización del dataset en un formato adecuado para su posterior análisis. Esta fase fue determinante para garantizar la calidad de los resultados.

Estudio de códigos en Python de machine learning

Una parte importante de la formación personal durante el TFG consistió en el estudio y análisis de diversos scripts en **Python** aplicados al machine learning. Se experimentó con diferentes modelos predictivos, evaluando su rendimiento y adaptándolos a las características del dataset. Esto supuso un aprendizaje progresivo en bibliotecas como **scikit-learn, XGBoost y pandas**, entre otras.

Desarrollo de la aplicación:

Figma

Para la fase de diseño se empleó Figma, lo que permitió crear prototipos de la aplicación, validar la usabilidad de la interfaz y definir el estilo visual del proyecto antes de proceder al desarrollo técnico.

MongoDB

Se diseñó e implementó una base de datos en MongoDB para almacenar y gestionar la información recopilada. La elección de esta tecnología permitió un manejo flexible de datos semiestructurados y facilitó la escalabilidad del proyecto.

Flask

Se empleó Flask para el desarrollo del backend, creando una API ligera y modular en Python. Su uso permitió definir endpoints para la gestión de usuarios y la conexión con los modelos de predicción, además de integrar de forma eficiente la base de datos en MongoDB y la interfaz en React.

React

En la parte del frontend se desarrolló una interfaz con React, orientada a la visualización de métricas y resultados de los modelos. Este trabajo implicó el diseño de componentes reutilizables, la integración con el backend y la optimización de la experiencia de usuario.

Memoria

Finalmente, se elaboró la **memoria del TFG**, donde se recogió todo el proceso seguido: desde la motivación inicial y el marco teórico, hasta la metodología aplicada, los resultados obtenidos y las conclusiones. Esta redacción requirió un esfuerzo de síntesis, rigor académico y una adecuada estructuración para reflejar de manera clara y coherente el trabajo realizado.

Capítulo 6 - English Version

6.1 Motivation

The increasing demands of elite football force teams to maximize every player's potential in order to achieve their sporting objectives. In a scenario where the margins between victory and defeat are minimal, having accurate and timely information about the physical condition of footballers becomes a decisive competitive advantage.

At the same time, the work of coaches and fitness staff is under constant pressure: they must plan strategies, manage workload, and make critical decisions within seconds, often with incomplete or unintuitive data. This complexity and urgency make it difficult to optimize both individual and collective performance throughout the season.

To carry out this project, we decided to rely on the STATSports platform, a company founded in 2007 in Newry, Northern Ireland, by Alan Clarke and Sean O'Connor. Its mission from the beginning has been to revolutionize the monitoring and analysis of sports performance through precision technology, offering real-time tools that help coaches, fitness staff, and players make better decisions.

The company carved out a place in the market thanks to its combination of high-precision GPS, inertial sensors, and intuitive software, which raised the standards of monitoring in elite sport. Soon, football and rugby teams, as well as national squads, began to trust its solutions.

Among its most notable milestones are:

- Partnerships with Premier League clubs and Ireland Rugby (2010–2014).
- International expansion with offices in the United States (2016), covering sports such as the NFL, NCAA, and MLS.
- The agreement with the English Football Association (FA) in 2018, which integrated its technology into all English national teams.
- The launch of the Apex Series in 2019, achieving centimeter-level precision.

- The creation of more accessible solutions such as Sonra Lite and Athlete Series, aimed also at universities, women's leagues, and amateur sports.

Today, STATSports is considered the global leader in athlete monitoring and GPS-based performance analysis, used by clubs such as Manchester City, Liverpool, Arsenal, and Juventus, as well as national teams including England, Argentina, the United States, and Australia.

With offices in Northern Ireland, London, Chicago, Florida, and Melbourne, the company provides global support and continues to innovate with cloud-based platforms, player-centered models, and monitoring tools for rehabilitation. Currently, its solutions are applied by thousands of teams worldwide, from grassroots academies to world champions.

6.2 Goals

The main objective is to develop an application that is easy to access and provides immediate consultation, which, based on physiological and performance data collected in short intervals during hydration breaks, time-outs, and half-time rests, predicts players' fatigue levels and provides the coaching staff with the objective information needed to optimize substitutions and tactical adjustments.

Another goal is to publish the resulting dataset in an open-access repository such as Kaggle. To date, there are very few public datasets that include fatigue and performance metrics in football, which makes it difficult to compare models and hinders collaborative progress in this field. Making this information available will facilitate future research and enhance the reproducibility of studies in sports analytics.

6.3 Work Plan

The development of the Final Degree Project (TFG) was structured following an agile methodology based on weekly sprints, each with clearly defined objectives (data analysis, module development, model testing, and documentation), along with internal checkpoints every week to ensure progress. In addition, formal reviews with our supervisor Ismael were conducted every two weeks, complemented by periodic validations with our coaches. This approach allowed us to iterate quickly, incorporate feedback from

different perspectives, and ensure both the quality of the work and compliance with academic deadlines.

Below, the main tasks are detailed by period:

Sprint 0 – October 2024

Activities: Research on sports platforms and APIs (StatsPerform, Mediacoach, SportMonks, Sportradar, FootyStats, STATSports Apex Coach Series).

Deliverable: List and summary of selected sources and APIs.

Sprint 0 bis – November–December 2024

Activities: Acquisition and configuration of the GPS vest (connectivity and calibration).

Installation and setup of the data capture application.

Detection and resolution of issues: synchronization, output formats, and reading stability.

Deliverable: Integration report of the vest and record of problems and solutions.

Sprint 1 – 12–18 February 2025

Activities: Presentation of initial CSV files (data from the vest).

Definition of test subjects (Daniel and Alejandro).

Definition of DSL, HSR, and HID metrics.

Deliverable: Base CSV and document of metrics to be used.

Sprint 2 – 19–25 February 2025

Activities: Literature review on distance metrics (Euclidean, Manhattan, Mahalanobis…).

Selection of algorithms (k-NN, Decision Tree, Random Forest, Multilayer Perceptron).

Deliverable: Comparative report of algorithms.

Sprint 3 – 26 February–4 March 2025

Activities: Development of scripts xlsxAcsv.py and corregirCSV.py.

Automatic calculation of Pearson coefficients and regression lines on DSL.

Deliverable: Data cleaning code and statistical analysis outputs.

Sprint 4 – 5–11 March 2025

Activities: Generation of covariance and correlation matrices.

Filtering of variables with correlation ≥ 0.65 with DSL.

Deliverable: Correlation visualizations.

Sprint 5 – 12–18 March 2025

Activities: Segmentation of data (5×5 and 15×15).

Exploratory normalization.

First test of Multilayer Perceptron.

Deliverable: Segmentation scripts and preliminary results.

Sprint 6 – 19–25 March 2025

Activities: Consolidation of cumulative and normalized CSV.

Cross-validation of the Multilayer Perceptron.

Deliverable: Final CSV and validation metrics (MAE, R^2).

Sprint 7 – 26 March–1 April 2025

Activities: Adjustments to segmentation and normalization.

Testing of Perceptron and Random Forest.

Deliverable: Comparative performance report of both models.

Sprint 8 – 2–8 April 2025

Activities: Completion of preprocessing (calories, impacts).

Final training of Random Forest ($R^2 \approx 0.90$).

Deliverable: Random Forest report with final metrics.

Sprint 9 – 9–15 April 2025

Activities: Generation of results reports per subject (Daniel Ortiz, Alejandro Nafría, combined dataset).

Selection and configuration of XGBRegressor and XGBClassifier for DSL.

Deliverable: XGBoost model proposal.

Sprint 10 – 16–22 April 2025

Activities: Update of CSV (conversion of hours to minutes, IDs, normalization of DSL).

Retraining and adjustment of models.

Deliverable: Revised final CSV and updated metrics.

Sprint 11 – 23–29 April 2025

Activities: Final comparative analysis (MLP vs. Random Forest vs. XGBoost).

Preparation of tables and graphs for the dissertation.

Deliverable: Draft of the "Model Development" chapter.

Sprint 12 – 30 April–6 May 2025

Activities: Writing and formatting of the dissertation.

Internal review of previous chapters.

Deliverable: Preliminary complete version of the dissertation.

Sprint 13 – 7–13 May 2025

Activities: Development of the interactive application (Streamlit/Node.js).

Installation of Node.js and MongoDB Compass.

Deliverable: Prototype of the web interface.

Sprint 14 – 14–20 May 2025

Activities: Implementation of the prediction endpoint (XGBoost API).

UX testing and parameter adjustments.

Deliverable: Functional real-time prediction demo.

Sprint 15 – 21–27 May 2025

Activities: User testing with coaches.

Collection of feedback and final corrections.

Deliverable: Usability report and improvement plan.

Sprint 16 – 28 May–3 June 2025

Activities: Writing of conclusions and future work.

Final formatting according to academic standards.

Deliverable: Draft of the dissertation for submission.

Sprint 17 – July–August 2025

Activities: Finalization of both the dissertation and the platform.

Deliverable: Final dissertation and MVP of the platform available, along with public dataset.

Bi-weekly reviews with the supervisor (Ismael): February 12, February 26, March 12, March 26, April 9, April 23, May 7, May 21, July 15, and August 15 — to present deliverables, resolve doubts, and adjust the plan

6.4 Conclusions and Future Work

The following section presents the conclusions of our project, evaluating both the initial approach and the results obtained, and proposing improvements and ideas for future versions of the application.

The objective of this Final Degree Project was to combine two of our main pillars —university training and sport—, taking as a starting point the work "Generation of a Machine Learning Tool to Support Athletes" by our colleagues Juan Israel Baroffi González and Javier Parra González in 2024, and adapting it to the football context.

From the beginning, we faced the usual challenges of machine learning projects. First, we had to filter and select the most relevant variables from the large volume of raw data available in our "device". Thanks to the development of the Pearson correlation

coefficient and correlation matrices, we were able to structure the information and reduce it to a fixed set of columns. After several months dedicated to the collection, cleaning, and organization of these data into our CSV file, we proceeded to train and evaluate different machine learning algorithms, achieving very satisfactory results.

On the other hand, we set as a secondary objective the publication of the training dataset, which we uploaded to the Kaggle platform and can be consulted at the following URL:
https://www.kaggle.com/datasets/alejandronafria/football-fatigue-metrics-dsl

As future lines of work, we identified several key limitations that we would like to overcome:

Sample temporality: a longer collection period would have allowed the use of alternative algorithms and comparison of their accuracy.

Expanding the evaluation sample: in an initial phase we tried to include Mario, a teammate of Alejandro from the second team, as a test subject. However, the logistics of lending him the sensorized vest (coordination of training sessions, synchronization of devices, and data cleaning) proved too complex.

For future iterations we plan to:

2.1 Incorporate additional vests or loan agreements to facilitate simultaneous access for several players.

2.2 Establish a standardized data collection protocol (schedules, export format, automatic filters) that minimizes post-processing time and ensures information quality.

2.3 Acquire a heart rate monitor strap: this will allow us to measure heart rate and variability more accurately during training and matches. By integrating it with our system, we will obtain more detailed load curves, improve the accuracy of predictive algorithms, and enrich performance reports for the coaching staff.

Technological and functional improvements for the system:

Development of a mobile application to facilitate both real-time visualization of metrics and data management by coaches.

Obtain the official STATSports API, which would allow us to access more precise and complete metrics than those obtained through manual exports.

Implement the option to manually add data into the application, useful for recording incidents, observations, or metrics not captured by the vest.

Train more complex machine learning models (deep neural networks, sequential models, etc.) to improve robustness and predictive capacity.

Enable the system to work directly with the database, instead of relying on manually added files, thereby increasing realism and scalability.

Incorporate the option to save training sessions and matches directly into the database, creating a structured session history.

Develop the possibility of visualizing graphs per session, either for a single player or for an entire group, enabling easier comparison of profiles.

Expand prediction capabilities to estimate metrics at different time horizons (60, 75 minutes, etc.) and not only at the end of the match or training session.

Extend predictions to new dimensions, such as accumulated workload or extra fatigue, providing greater value to the coaching staff.

Improve the visual aspect of the platform, for example by incorporating color ranges that highlight critical indicators such as DSL after prediction, facilitating quick and intuitive interpretation.

Create an automated tool for generating CSVs in the appropriate format from the data obtained via the API, reducing manual data preparation work and streamlining the data ingestion flow into the system.

Propose as a future line of work the deployment of the platform using containers (Docker), which would allow greater portability, scalability, and ease of maintenance in production environments.

These initiatives will strengthen the project's ability to provide more reliable predictions and real added value to the sports decision-making process.

Despite these limitations, we are very satisfied with the prototype developed and are confident that the proposals described here will serve as a springboard for future improvements. The test results demonstrate that the model provides predictions with a satisfactory level of accuracy.

BIBLIOGRAFÍA

[1]: Microsoft Prensa, Beyond Stats. URL: https://news.microsoft.com/es-es/2021/10/06/laliga-y-microsoft-presentan-beyond-stats-un-proyecto-de-analisis-futbolistico-avanzado-que-profundiza-en-el-juego-de-cada-equipo/ (Accedido el 10/09/2025)

[2]:La Liga, Beyond Stats, URL: https://www.laliga.com/en-GB/beyondstats (Accedido el 10/09/2025)

[3]: Alejandro Nafria, Futbol fatigue metrics, URL:

https://www.kaggle.com/datasets/alejandronafria/football-fatigue-metrics-dsl (Accedido el 10/09/2025)

[4] Rodrıguez, M. M. Modelización estadıstica en pruebas de esfuerzo y rendimiento deportivo URL: http://eio.usc.es/pub/mte/descargas/ProyectosFinMaster/Proyecto_1480.pdf (Accedido el 10/09/2025)

[5] Paulis, J. C., & Mendo, A. H. (2000). Análisis secuencial en el fútbol de rendimiento. *Psicothema*, 117-121. URL: https://reunido.uniovi.es/index.php/PST/article/view/7656/7520 (Accedido el 10/09/2025)

[6] Jiménez-Reyes, P., Cuadrado-Peñafiel, V., & González-Badillo, J. J. (2011). Análisis de variables medidas en salto vertical relacionadas con el rendimiento deportivo y su aplicación al entrenamiento. *Cultura, Ciencia y Deporte*, 6(17), 113-119. URL https://www.redalyc.org/pdf/1630/163022532005.pdf (Accedido el 10/09/2025)

[7] De Bortoli, A. L., De Bortoli, R., & Márquez, S. (2001). Utilización de coeficientes ofensivos para el análisis del rendimiento deportivo en el fútbol sala. *Motricidad. European Journal of Human Movement*, *7*, 7-17. URL: https://www.redalyc.org/pdf/1630/163022532005.pdf (Accedido el 10/09/2025)

[8] Juan Israel Baroffi González & Javier Parra González, (2024).Generación de una herramienta de machine learning para apoyar a depertistas URL: https://docta.ucm.es/entities/publication/8397d87a-5aea-4714-a17f-55bfbe66caae (Accedido el 10/09/2025)

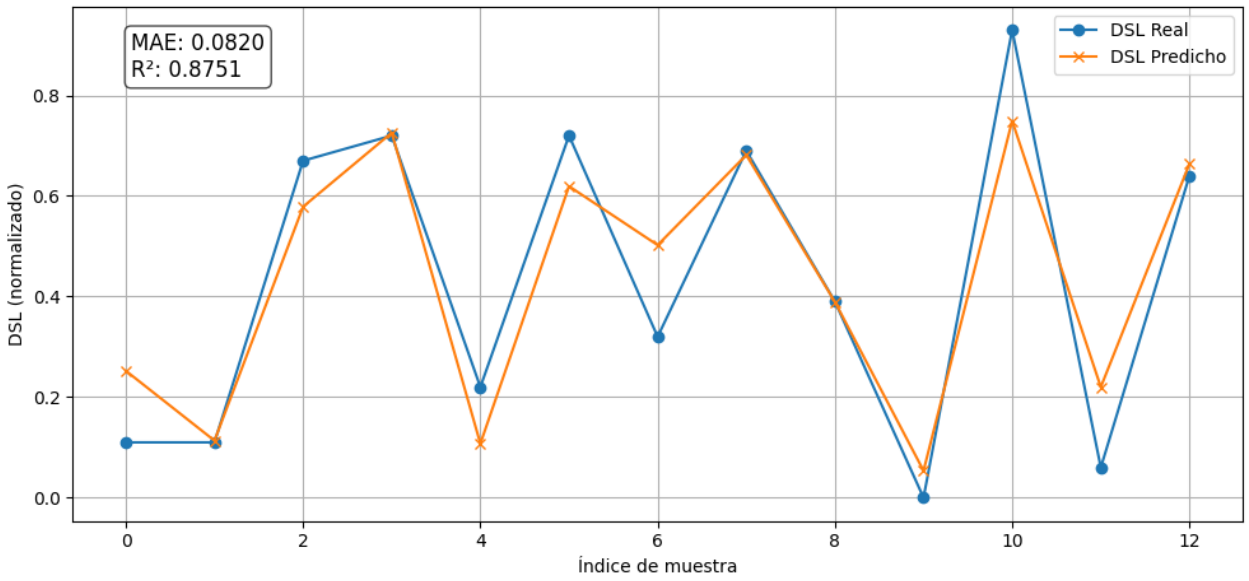
[9] CHALECOS GPS DE ENTRENAMIENTO, URL: https://www.livingfootball.es/comprar-chalecos-gps-entrenamiento/ (Accedido el 10/09/2025)

APÉNDICES

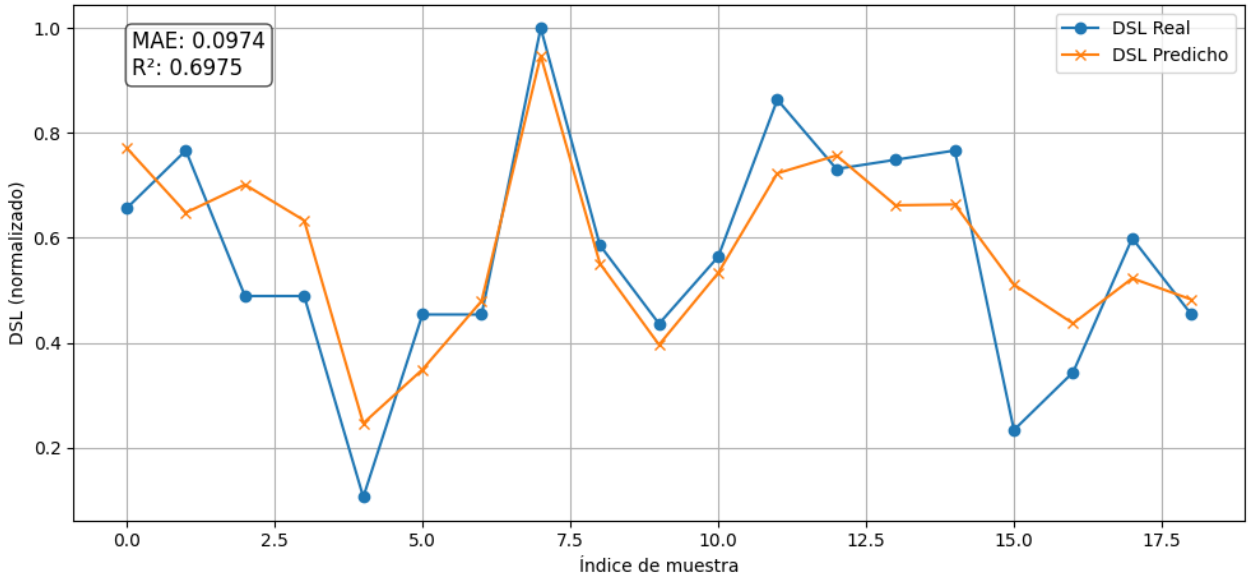
Apéndice A - Comparaciones DSL PREDICHO – DSL REAL

Perceptrón

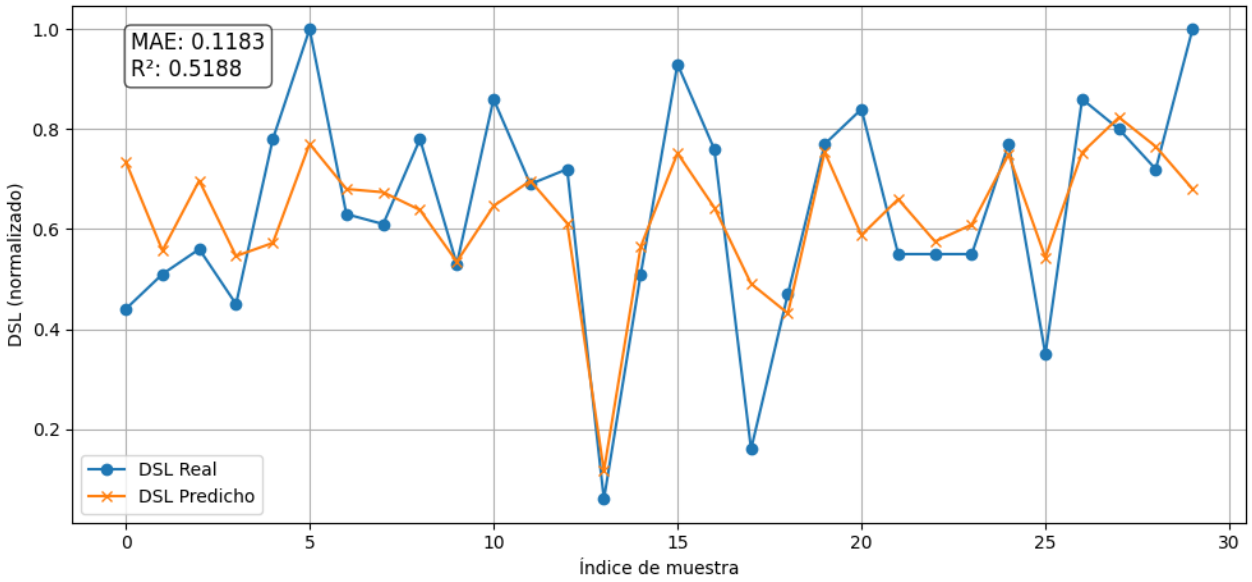
CSV Daniel



CSV Alejandro

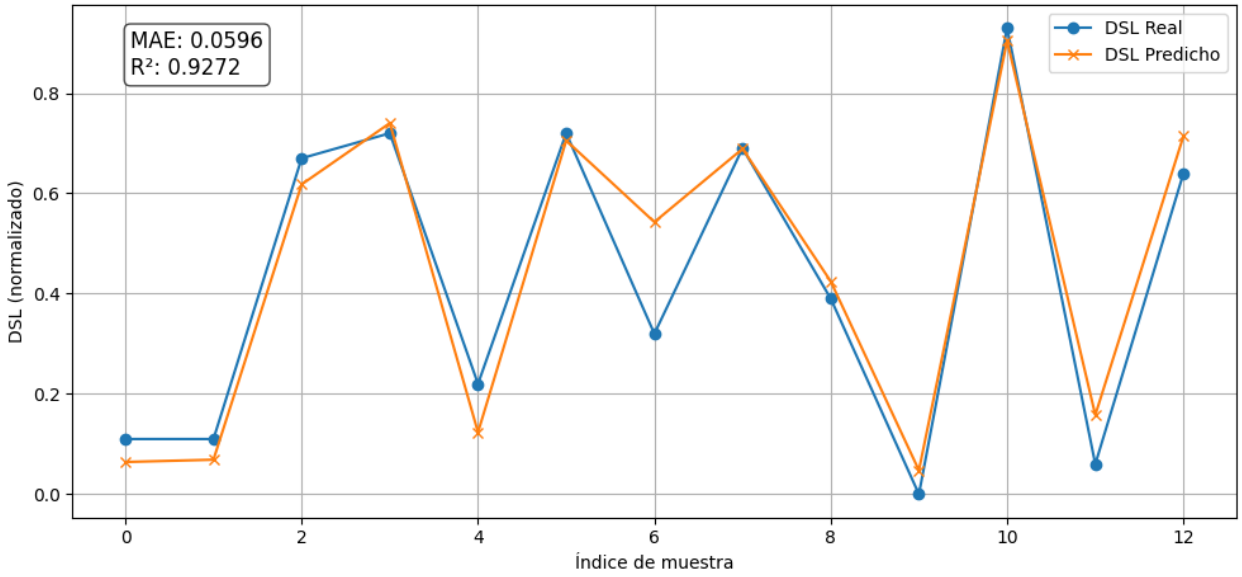


CSV Unido

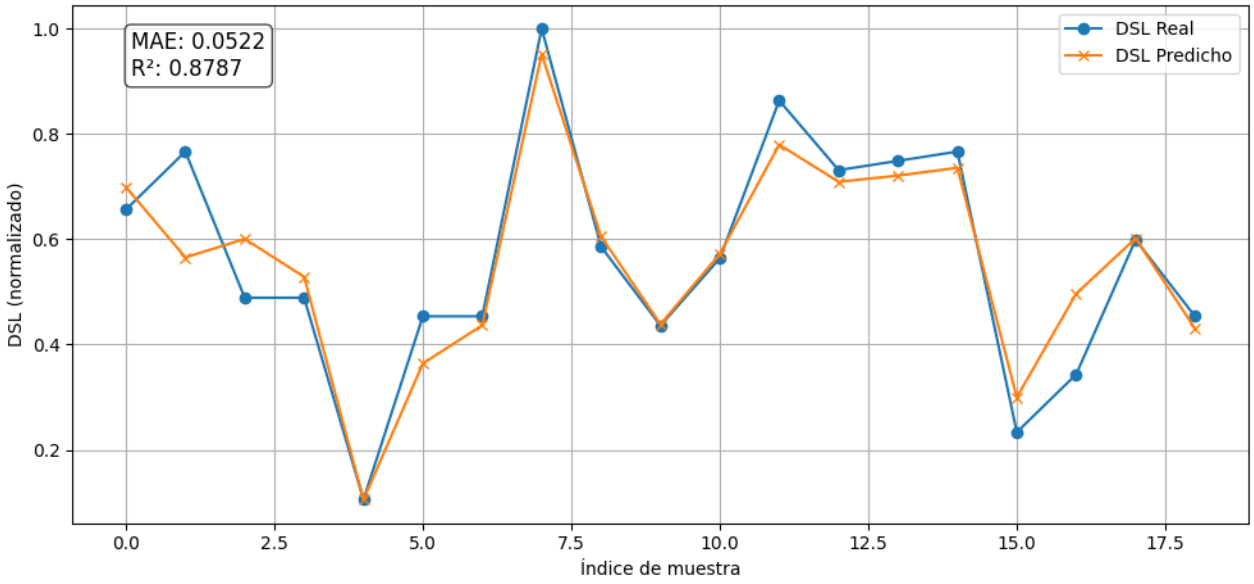


RandomForest

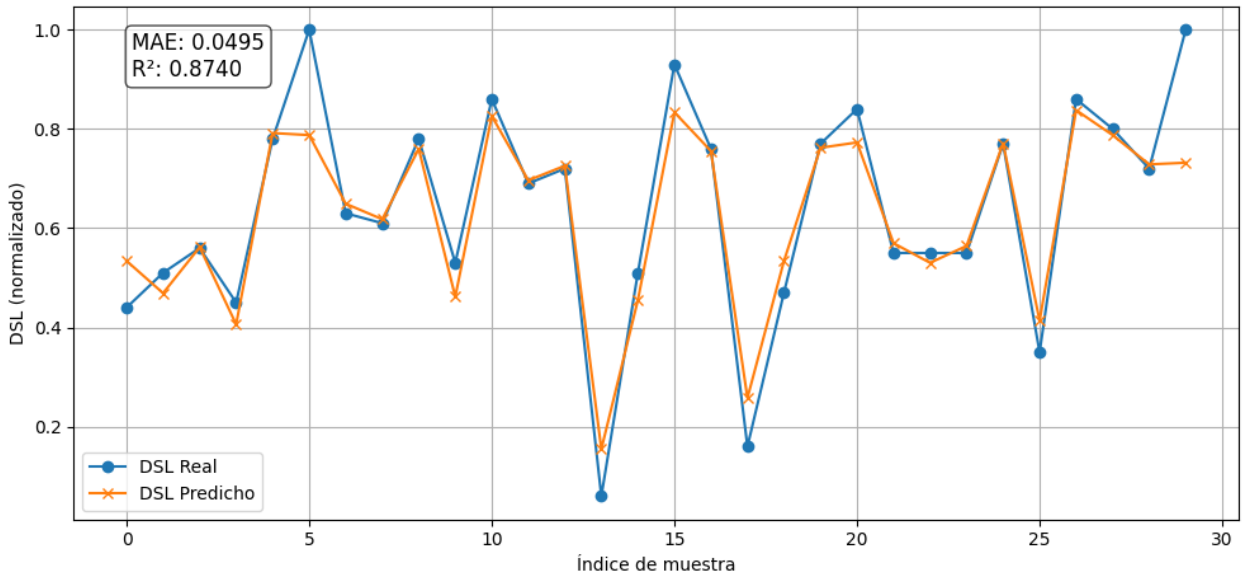
CSV Daniel



CSV Alejandro

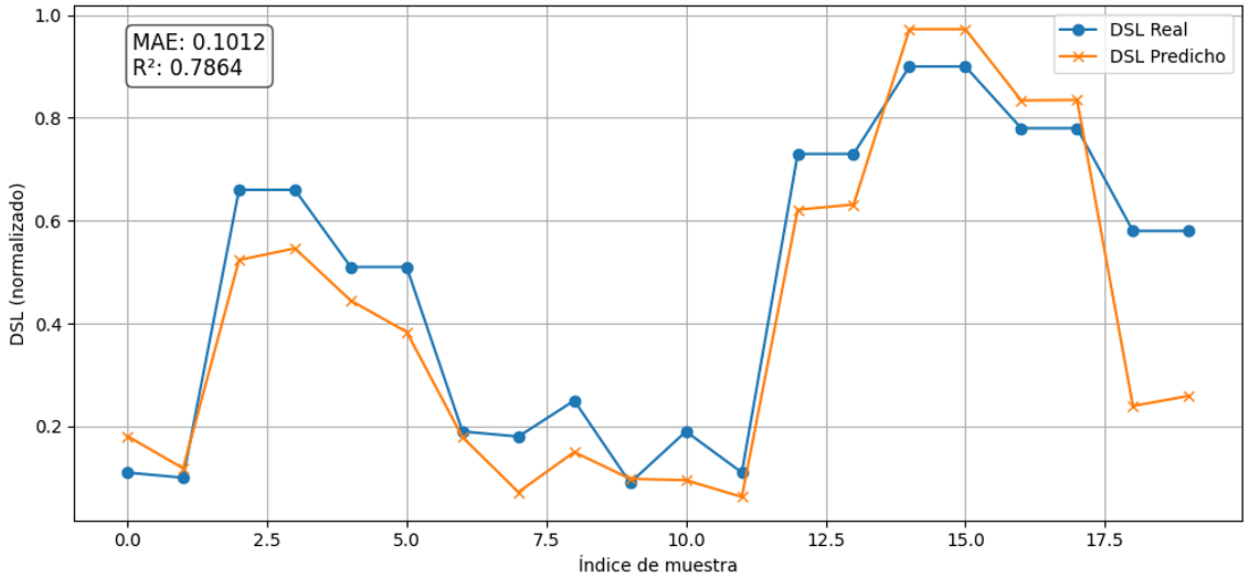


CSV Unido

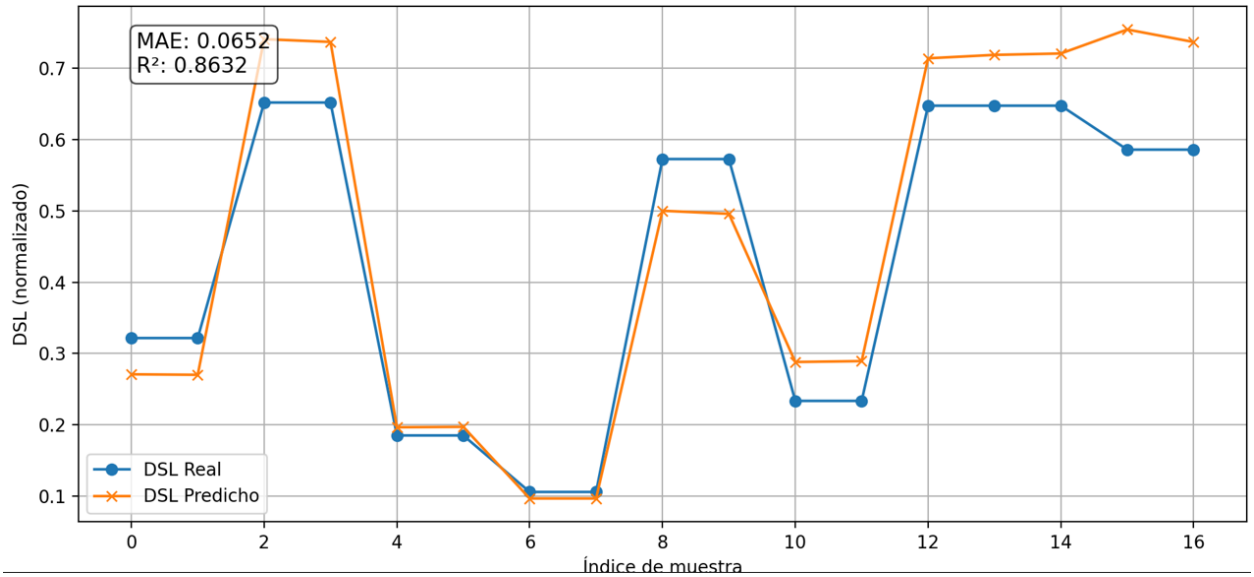


XGBoost

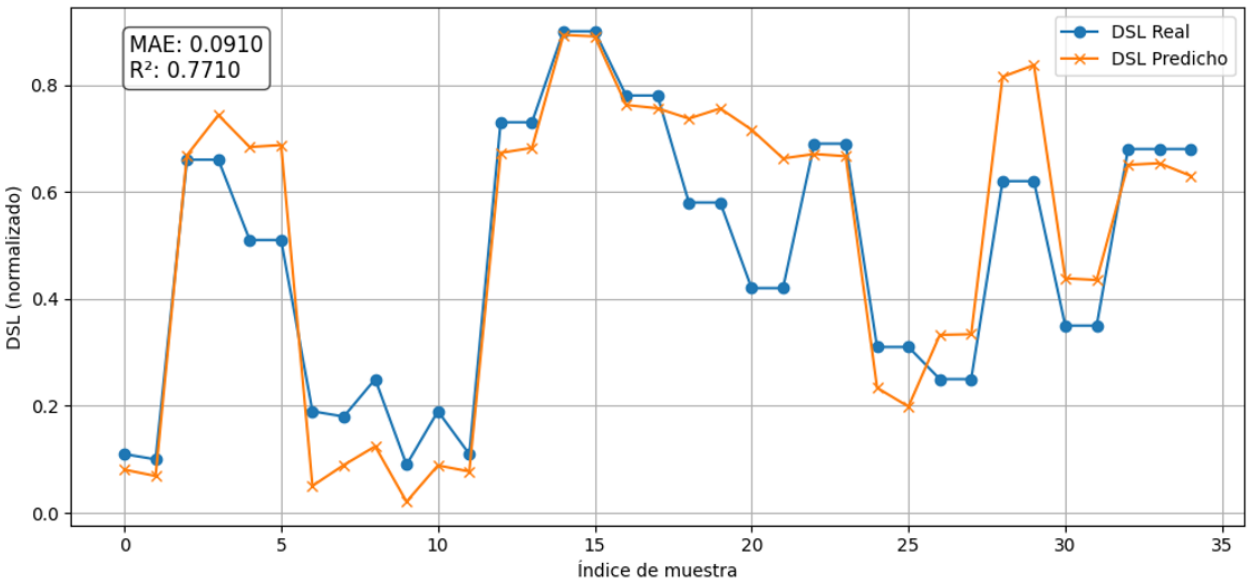
CSV Daniel



CSV Alejandro



CSV Unido



Apéndice B - Algoritmos empleados de Machine-Learning para las gráficas anteriores

Perceptrón

```python
import pandas as pd
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# Cargar los datasets
df_train = pd.read_csv("Alcex_Desglose15_Unido45_Normalizado_MinMax_DatosEntrenamiento.csv")
df_pred = pd.read_csv("Alcex_Desglose15_Unido45_Normalizado_MinMax_DatosParaPredecir.csv")

# Columnas a excluir como features
excluded_columns = {'ID', 'Segmento1', 'Segmento2', 'Segmento3', 'SubSegmento', 'TiempoRestante',
                    'HSR por minuto ', 'DSL'}

# Variables independientes y dependiente
X_train = df_train.drop(columns=excluded_columns, errors='ignore')
y_train = df_train['DSL']

X_pred = df_pred.drop(columns=excluded_columns, errors='ignore')
y_real = df_pred['DSL']  # Valores reales para comparación

# Crear y entrenar el perceptrón
mlp = MLPRegressor(hidden_layer_sizes=(64, 32), max_iter=1000, random_state=42)
mlp.fit(X_train, y_train)

# Predicción sobre el dataset de predicción
y_pred = mlp.predict(X_pred)

# Métricas de evaluación
mae = mean_absolute_error(y_real, y_pred)
r2 = r2_score(y_real, y_pred)

print(f"Error absoluto medio (MAE): {mae:.4f}")
print(f"Coeficiente de determinación (R²): {r2:.4f}")

# Gráfica: comparación entre DSL real y predicho
plt.figure(figsize=(10, 5))
plt.plot(y_real.values, label="DSL Real", marker='o')
plt.plot(y_pred, label="DSL Predicho", marker='x')
plt.title("Perceptron - Predicción sobre nuevos datos")
plt.xlabel("Índice de muestra")
plt.ylabel("DSL (normalizado)")
plt.legend()
plt.grid(True)

# Añadir MAE y R² como texto en la gráfica
plt.text(0.05, 0.95, f"MAE: {mae:.4f}\nR²: {r2:.4f}",
         transform=plt.gca().transAxes,
         fontsize=12,
         verticalalignment='top',
         bbox=dict(boxstyle="round", facecolor="white", alpha=0.7))

plt.tight_layout()
plt.show()
```

RandomForest

```python
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# Cargar los datasets separados por ID
df_train = pd.read_csv("Alcex_Desglose15_Unido45_Normalizado_MinMax_DatosEntrenamiento.csv")
df_pred = pd.read_csv("Alcex_Desglose15_Unido45_Normalizado_MinMax_DatosParaPredecir.csv")

# Columnas a excluir como features
excluded_columns = {
    'ID', 'Segmento1', 'Segmento2', 'Segmento3', 'SubSegmento',
    'TiempoRestante', 'Step Balance (L)', 'Step Balance (R)', 'HSR por minuto ', 'DSL'
}

# Variables independientes (X) y dependiente (y)
X_train = df_train.drop(columns=excluded_columns, errors='ignore')
y_train = df_train['DSL']
X_pred = df_pred.drop(columns=excluded_columns, errors='ignore')
y_real = df_pred['DSL']

# Crear y entrenar el modelo Random Forest
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Predicción sobre el grupo de prueba
y_pred = rf.predict(X_pred)

# Métricas de evaluación
mae = mean_absolute_error(y_real, y_pred)
r2 = r2_score(y_real, y_pred)

print(f"Error absoluto medio (MAE): {mae:.4f}")
print(f"Coeficiente de determinación (R²): {r2:.4f}")

# Gráfica: comparación entre DSL real y predicho
plt.figure(figsize=(10, 5))
plt.plot(y_real.values, label="DSL Real", marker='o')
plt.plot(y_pred, label="DSL Predicho", marker='x')
plt.title("Random Forest - Predicción sobre nuevos datos (ID ≥ 29)")
plt.xlabel("Índice de muestra")
plt.ylabel("DSL (normalizado)")
plt.legend()
plt.grid(True)

# Añadir MAE y R² como texto en la gráfica
plt.text(0.05, 0.95, f"MAE: {mae:.4f}\nR²: {r2:.4f}",
         transform=plt.gca().transAxes,
         fontsize=12,
         verticalalignment='top',
         bbox=dict(boxstyle="round", facecolor="white", alpha=0.7))

plt.tight_layout()
plt.show()

# Importancia de las variables
importances = pd.Series(rf.feature_importances_, index=X_train.columns)
top_importances = importances.sort_values(ascending=False).head(10)

print("\nTop 10 variables más importantes:")
print(top_importances)
```

XGBoost

```python
import pandas as pd
from xgboost import XGBRegressor
from sklearn.metrics import mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# Cargar los datasets
df_train = pd.read_csv("Alecx_Desglose15_Unido45_Normalizado_MinMax_DatosEntrenamiento.csv")
df_pred = pd.read_csv("Alecx_Desglose15_Unido45_Normalizado_MinMax_DatosParaPredecir.csv")

# Columnas a excluir como features
excluded_columns = {'ID', 'Segmento1', 'Segmento2', 'Segmento3', 'SubSegmento', 'TiempoRestante',
                    'HSR por minuto ', 'DSL'}

# Variables independientes y dependiente
X_train = df_train.drop(columns=excluded_columns, errors='ignore')
y_train = df_train['DSL']

X_pred = df_pred.drop(columns=excluded_columns, errors='ignore')
y_real = df_pred['DSL']  # Valores reales para comparación

# Crear y entrenar el modelo XGBoost
xgb = XGBRegressor(objective='reg:squarederror', n_estimators=100, random_state=42)
xgb.fit(X_train, y_train)

# Predicción sobre el dataset de predicción
y_pred = xgb.predict(X_pred)

# Métricas de evaluación
mae = mean_absolute_error(y_real, y_pred)
r2 = r2_score(y_real, y_pred)

print(f"Error absoluto medio (MAE): {mae:.4f}")
print(f"Coeficiente de determinación (R²): {r2:.4f}")

# Gráfica: comparación entre DSL real y predicho
plt.figure(figsize=(10, 5))
plt.plot(y_real.values, label="DSL Real", marker='o')
plt.plot(y_pred, label="DSL Predicho", marker='x')
plt.title("XGBoost - Predicción sobre nuevos datos")
plt.xlabel("Índice de muestra")
plt.ylabel("DSL (normalizado)")
plt.legend()
plt.grid(True)

# Añadir MAE y R² como texto en la gráfica
plt.text(0.05, 0.95, f"MAE: {mae:.4f}\nR²: {r2:.4f}",
         transform=plt.gca().transAxes,
         fontsize=12,
         verticalalignment='top',
         bbox=dict(boxstyle="round", facecolor="white", alpha=0.7))

plt.tight_layout()
plt.show()

```

Apéndice C - Algoritmos empleados de Machine-Learning para la predicción en la plataforma

Perceptrón

```python
import os
import numpy as np
import pandas as pd
from sklearn.model_selection import KFold, cross_validate
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import MinMaxScaler
from sklearn.neural_network import MLPRegressor
from sklearn.compose import TransformedTargetRegressor

class Perceptron:
    def predict(self, file, fileFormat, mode, user_id, k = 5):
        # 1) Cargar TRAIN según 'mode'
        base_path = os.path.dirname(os.path.abspath(__file__))  # Carpeta donde está este script
        if mode and user_id == 2:
            csv_path = os.path.join(base_path, "..", "resources", "Danii_Desglose15_Unido45_plataforma.csv")
        elif mode and user_id == 3:
            csv_path = os.path.join(base_path, "..", "resources", "Aleex_Desglose15_Unido45_plataforma.csv")
        else:
            csv_path = os.path.join(base_path, "..", "resources", "DaniiAleex_Desglose15_Unido45_plataforma.csv")

        df_train = pd.read_csv(csv_path)

        # 2) Cargar PRED según 'fileFormat'
        if fileFormat == ".csv":
            df_pred = pd.read_csv(file)
        elif fileFormat == ".xlsx":
            df_pred = pd.read_excel(file)
        else:
            raise ValueError("fileFormat must be '.csv' or '.xlsx'")

        # Limpieza de nombres de columnas (por si hay espacios/variantes)
        #df_train.columns = df_train.columns.str.strip()
        #df_pred.columns  = df_pred.columns.str.strip()

        # 3) Definir columnas excluidas
        excluded_columns = {
            'ID', 'Segmento1', 'Segmento2', 'Segmento3', 'SubSegmento',
            'TiempoRestante', 'HSR por minuto', 'DSL'
        }

        # 4) Preparar X/y (train) y X (pred)
        X_train = df_train.drop(columns=excluded_columns, errors='ignore')
        y_train = df_train['DSL']

        X_pred = df_pred.drop(columns=excluded_columns, errors='ignore')

        # Asegurar tipo numérico (convierte no numéricas a NaN)
        #X_train = X_train.apply(pd.to_numeric, errors='coerce')
        #X_pred  = X_pred.apply(pd.to_numeric, errors='coerce')


        # Alinear columnas de pred a las de train
        #X_pred = X_pred.reindex(columns=X_train.columns)

        # --- 4) Modelo: MinMax en X + MLP, y MinMax en y con TTR ---
        reg_x = Pipeline(steps=[
            ("scaler_x", MinMaxScaler()),
            ("mlp", MLPRegressor(hidden_layer_sizes=(64, 32),
                                 max_iter=1000, random_state=42))
        ])
        # Transforma y con MinMax y destransforma al predecir
        model = TransformedTargetRegressor(regressor=reg_x,
                                           transformer=MinMaxScaler())

        # --- 5) Fiabilidad global con K-Fold (MAE/R2 media ± std) ---
        cv_metrics = None
        n_samples = len(y_train)
        effective_k = min(max(2, k), n_samples)  # k en [2, n]
        if effective_k >= 2:
            cv = KFold(n_splits=effective_k, shuffle=True, random_state=42)
            scores = cross_validate(
                model, X_train, y_train,
                scoring={'mae': 'neg_mean_absolute_error', 'r2': 'r2'},
                cv=cv, n_jobs=1, return_train_score=False
            )
            mae_vals = -scores['test_mae']  # convertir a positivo
            r2_vals  = scores['test_r2']
            cv_metrics = {
                "mae_mean": float(mae_vals.mean()),
                "mae_std":  float(mae_vals.std(ddof=1)),
                "r2_mean":  float(r2_vals.mean()),
                "r2_std":   float(r2_vals.std(ddof=1)),
                "k":        int(effective_k)
            }

        # --- 6) Entrenar en TODO el train y predecir ---
        model.fit(X_train, y_train)
        y_pred_original = model.predict(X_pred)  # ya en escala original (desnormalizado)

        y_scaler = model.transformer_          # MinMaxScaler ajustado con y_train
        y_pred_norm = y_scaler.transform(y_pred_original.reshape(-1, 1)).ravel()

        # --- 7) Devolver resultados ---
        result = {
            "cv_reliability": cv_metrics,          # media±std MAE/R2 (global)
            "dsl_pred": y_pred_original.tolist(),  # escala original
            "dsl_pred_norm_0_1": y_pred_norm.tolist()  # normalizado 0-1 respecto a histórico
        }

        return result
```

RandomForest

```python
import os
import numpy as np
import pandas as pd
from sklearn.model_selection import KFold, cross_validate
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import MinMaxScaler
from sklearn.compose import TransformedTargetRegressor
from sklearn.ensemble import RandomForestRegressor

class RandomForest:
    def predict(self, file, fileFormat, mode, user_id, k=5):
        # --- 1) Cargar TRAIN según 'mode' ---
        base_path = os.path.dirname(os.path.abspath(__file__))
        if mode and user_id == 2:
            csv_path = os.path.join(base_path, "..", "resources", "Danii_Desglose15_Unido45_plataforma.csv")
        elif mode and user_id == 3:
            csv_path = os.path.join(base_path, "..", "resources", "Aleex_Desglose15_Unido45_plataforma.csv")
        else:
            csv_path = os.path.join(base_path, "..", "resources", "DaniiAleex_Desglose15_Unido45_plataforma.csv")

        df_train = pd.read_csv(csv_path)

        # --- 2) Cargar PRED según 'fileFormat' ---
        if fileFormat == ".csv":
            df_pred = pd.read_csv(file)
        elif fileFormat == ".xlsx":
            df_pred = pd.read_excel(file)
        else:
            raise ValueError("fileFormat must be '.csv' or '.xlsx'")

        #Limpieza de nombres de columnas (evita espacios raros)
        #df_train.columns = df_train.columns.str.strip()
        #df_pred.columns  = df_pred.columns.str.strip()

        # --- 3) Preparar X/y y excluir columnas no predictoras ---
        excluded_columns = {
            'ID','Segmento1','Segmento2','Segmento3','SubSegmento',
            'TiempoRestante','HSR por minuto','DSL'
        }
        """
        X_train = df_train.drop(columns=[c for c in df_train.columns if c in excluded_columns], errors='ignore')
        y_train = df_train['DSL']

        X_pred = df_pred.drop(columns=[c for c in df_pred.columns if c in excluded_columns], errors='ignore')
        """

        X_train = df_train.drop(columns=excluded_columns, errors='ignore')
        y_train = df_train['DSL']

        X_pred = df_pred.drop(columns=excluded_columns, errors='ignore')

        # Asegurar numérico (texto -> NaN) y alinear columnas
        #X_train = X_train.apply(pd.to_numeric, errors='coerce')
        #X_pred  = X_pred.apply(pd.to_numeric, errors='coerce')
        #X_pred  = X_pred.reindex(columns=X_train.columns)

        # --- 4) Modelo: MinMax en X + RandomForest, y MinMax en y con TTR ---
        reg_x = Pipeline(steps=[
            ("scaler_x", MinMaxScaler()),
            ("rf", RandomForestRegressor(
                n_estimators=300,
                random_state=42,
                n_jobs=1,        # evita multiprocessing dentro del servidor
                max_depth=None,  # puedes tunear si quieres
            ))
        ])
        model = TransformedTargetRegressor(
            regressor=reg_x,
            transformer=MinMaxScaler()  # normaliza y y desnormaliza al predecir
        )

        # --- 5) Fiabilidad global con K-Fold (MAE/R2 media ± std) ---
        cv_metrics = None
        n_samples = len(y_train)
        effective_k = min(max(2, k), n_samples)  # k en [2, n]
        if effective_k >= 2:
            cv = KFold(n_splits=effective_k, shuffle=True, random_state=42)
            scores = cross_validate(
                model, X_train, y_train,
                scoring={'mae': 'neg_mean_absolute_error', 'r2': 'r2'},
                cv=cv, n_jobs=1, return_train_score=False  # n_jobs=1 en servidor
            )
            mae_vals = -scores['test_mae']
            r2_vals  = scores['test_r2']
            cv_metrics = {
                "mae_mean": float(mae_vals.mean()),
                "mae_std":  float(mae_vals.std(ddof=1)),
                "r2_mean":  float(r2_vals.mean()),
                "r2_std":   float(r2_vals.std(ddof=1)),
                "k":        int(effective_k)
            }

        # --- 6) Entrenar en TODO el train y predecir ---
        model.fit(X_train, y_train)
        y_pred_original = model.predict(X_pred)  # escala original (desnormalizado)

        y_scaler = model.transformer_
        y_pred_norm = y_scaler.transform(y_pred_original.reshape(-1, 1)).ravel()

        # --- 7) Resultado listo para JSON ---
        result = {
            "cv_reliability": cv_metrics,                # MAE/R2 media±std
            "dsl_pred": y_pred_original.tolist(),        # escala original
            "dsl_pred_norm_0_1": y_pred_norm.tolist(),   # normalizado 0-1
        }

        return result
```

XGBoost

```python
import os
import numpy as np
import pandas as pd
from sklearn.model_selection import KFold, cross_validate
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import MinMaxScaler
from sklearn.impute import SimpleImputer
from sklearn.compose import TransformedTargetRegressor
import xgboost as xgb

class XGBRegressor:
    def predict(self, file, fileFormat, mode, user_id, k=5):
        # --- 1) Cargar TRAIN según 'mode' ---
        base_path = os.path.dirname(os.path.abspath(__file__))
        if mode and user_id == 2:
            csv_path = os.path.join(base_path, "..", "resources", "Danii_Desglose15_Unido45_plataforma.csv")
        elif mode and user_id == 3:
            csv_path = os.path.join(base_path, "..", "resources", "Aleex_Desglose15_Unido45_plataforma.csv")
        else:
            csv_path = os.path.join(base_path, "..", "resources", "DaniiAleex_Desglose15_Unido45_plataforma.csv")

        df_train = pd.read_csv(csv_path)

        # --- 2) Cargar PRED según 'fileFormat' ---
        if fileFormat == ".csv":
            df_pred = pd.read_csv(file)
        elif fileFormat == ".xlsx":
            df_pred = pd.read_excel(file)
        else:
            raise ValueError("fileFormat must be '.csv' or '.xlsx'")

        # Limpieza nombres (evita espacios raros)
        #df_train.columns = df_train.columns.str.strip()
        #df_pred.columns  = df_pred.columns.str.strip()

        # --- 3) Preparar X/y y excluir columnas no predictoras ---
        excluded_columns = {
            'ID','Segmento1','Segmento2','Segmento3','SubSegmento',
            'TiempoRestante','HSR por minuto','DSL'
        }

        """
        X_train = df_train.drop(columns=[c for c in df_train.columns if c in excluded_columns], errors='ignore')
        y_train = df_train['DSL']

        X_pred = df_pred.drop(columns=[c for c in df_pred.columns if c in excluded_columns], errors='ignore')
        """

        X_train = df_train.drop(columns=excluded_columns, errors='ignore')
        y_train = df_train['DSL']

        X_pred = df_pred.drop(columns=excluded_columns, errors='ignore')

        # Numérico + alinear columnas
        #X_train = X_train.apply(pd.to_numeric, errors='coerce')
        #X_pred  = X_pred.apply(pd.to_numeric, errors='coerce')
        #X_pred  = X_pred.reindex(columns=X_train.columns)

        # --- 4) Modelo: Imputer + MinMax en X + XGB, y MinMax en y con TTR ---
        reg_x = Pipeline(steps=[
            ("imputer", SimpleImputer(strategy="median")),
            ("scaler_x", MinMaxScaler()),
            ("xgb", xgb.XGBRegressor(
                objective="reg:squarederror",
                n_estimators=400,
                learning_rate=0.05,
                max_depth=5,
                subsample=0.9,
                colsample_bytree=0.9,
                reg_lambda=1.0,
                random_state=42,
                n_jobs=1  # evita multiprocessing dentro del servidor
            ))
        ])
        model = TransformedTargetRegressor(
            regressor=reg_x,
            transformer=MinMaxScaler()  # normaliza y y desnormaliza al predecir
        )

        # --- 5) Fiabilidad global con K-Fold (MAE/R2 media ± std) ---
        cv_metrics = None
        n_samples = len(y_train)
        effective_k = min(max(2, k), n_samples)
        if effective_k >= 2:
            cv = KFold(n_splits=effective_k, shuffle=True, random_state=42)
            scores = cross_validate(
                model, X_train, y_train,
                scoring={'mae': 'neg_mean_absolute_error', 'r2': 'r2'},
                cv=cv, n_jobs=1, return_train_score=False
            )
            mae_vals = -scores['test_mae']
            r2_vals  = scores['test_r2']
            cv_metrics = {
                "mae_mean": float(mae_vals.mean()),
                "mae_std":  float(mae_vals.std(ddof=1)),
                "r2_mean":  float(r2_vals.mean()),
                "r2_std":   float(r2_vals.std(ddof=1)),
                "k":        int(effective_k)
            }

        # --- 6) Entrenar en TODO el train y predecir ---
        model.fit(X_train, y_train)
        y_pred_original = model.predict(X_pred)  # escala original (desnormalizado)

        # También devolver versión 0–1 con el MISMO scaler de y del TTR
        y_scaler = model.transformer_
        y_pred_norm = y_scaler.transform(y_pred_original.reshape(-1, 1)).ravel()

        # --- 7) Resultado listo para JSON ---
        result = {
            "cv_reliability": cv_metrics,              # MAE/R2 media±std
            "dsl_pred": y_pred_original.tolist(),      # escala original
            "dsl_pred_norm_0_1": y_pred_norm.tolist(),  # normalizado 0–1
        }

        return result
```

Apéndice D – Coeficientes correlación Pearson Febrero 2025

Código C++ empleado

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <cmath>

using namespace std;

double calcularPearson(const vector<double>& x, const vector<double>& y) {
    int n = x.size();
    double sumX = 0, sumY = 0, sumXY = 0, sumX2 = 0, sumY2 = 0;

    for (int i = 0; i < n; i++) {
        sumX += x[i];
        sumY += y[i];
        sumXY += x[i] * y[i];
        sumX2 += x[i] * x[i];
        sumY2 += y[i] * y[i];
    }

    double numerator = (n * sumXY) - (sumX * sumY);
    double denominator = sqrt((n * sumX2 - sumX * sumX) * (n * sumY2 - sumY * sumY));

    if (denominator == 0) return 0; // Evitar división por cero
    return numerator / denominator;
}

int main() {
    string filename;
    cout << "Elija un archivo para abrir:\n";
    cout << "1. datosAlex\n";
    cout << "2. datosDani11\n";
    cout << "3. datosDaniFutsal\n";
    cout << "4. datosDaniTotal\n";
    cout << "5. datosMario\n";

    int choice;
    cin >> choice;

    switch (choice) {
    case 1: filename = "datosAlex.txt"; break;
    case 2: filename = "datosDani11.txt"; break;
    case 3: filename = "datosDaniFutsal.txt"; break;
    case 4: filename = "datosDaniTotal.txt"; break;
    case 5: filename = "datosMario.txt"; break;
    default:
        cout << "Opción no válida." << endl;
        return 1;
    }

    ifstream inputFile(filename);
    if (!inputFile) {
        cout << "Error al abrir el archivo." << endl;
        return 1;
    }

    int X;
    inputFile >> X; // Leer el tamaño de los arrays

    vector<vector<double>> arrays(11, vector<double>(X)); // 11 arrays independientes
    vector<string> labels(11); // Array para almacenar las etiquetas

    // Leer los 11 arrays
    for (int i = 0; i < 11; i++) {
        inputFile >> labels[i]; // Guardar la palabra antes del array
        for (int j = 0; j < X; j++) {
            inputFile >> arrays[i][j];
        }
    }

    inputFile.close();

    string outputFilename = "resultadoPearson_" + filename;
    ofstream outputFile(outputFilename);
    if (!outputFile) {
        cerr << "Error al crear el archivo de salida." << endl;
        return 1;
    }

    // Calcular y mostrar el coeficiente de correlación de Pearson
    cout << "\nCoeficiente de correlacion de Pearson con " << labels[10] << ":\n";
    outputFile << "Coeficiente de correlacion de Pearson con " << labels[10] << ":\n";
    for (int i = 0; i < 10; i++) {
        double coef = calcularPearson(arrays[i], arrays[10]);
        cout << labels[i] << " - " << labels[10] << ": " << coef << endl;
        outputFile << labels[i] << " - " << labels[10] << ": " << coef << endl;
    }

    outputFile.close();
    cout << "Resultados guardados en " << outputFilename << endl;

    return 0;
}
```

Resultado Pearson datos Alejandro entrenamiento Fútbol Sala

```
Coeficiente de correlación de Pearson con DSL:
DistTotal - DSL: 0.640295
CorrerAltVel - DSL: 0.448587
DistPorMin - DSL: 0.681302
Impactos - DSL: 0.501675
MaxVelocidad - DSL: 0.797962
DistanciaAltaVelocidad - DSL: 0.558467
NumSprints - DSL: 0.39155
DistSprints - DSL: 0.421167
Aceleraciones - DSL: 0.343296
Desaceleraciones - DSL: 0.380731
```

Resultado Pearson datos Daniel partidos Fútbol 11

```
Coeficiente de correlación de Pearson con DSL:
DistTotal - DSL: 0.544299
CorrerAltVel - DSL: 0.384052
DistPorMin - DSL: 0.688885
Impactos: - DSL: 0.857434
MaxVelocidad - DSL: 0.621193
DistanciaAltaVelocidad - DSL: 0.63769
NumSprints - DSL: 0.320874
DistSprints - DSL: 0.229425
Aceleraciones - DSL: 0.00363334
Desaceleraciones - DSL: 0.430054
```

Resultado Pearson datos Daniel partidos Fútbol Sala

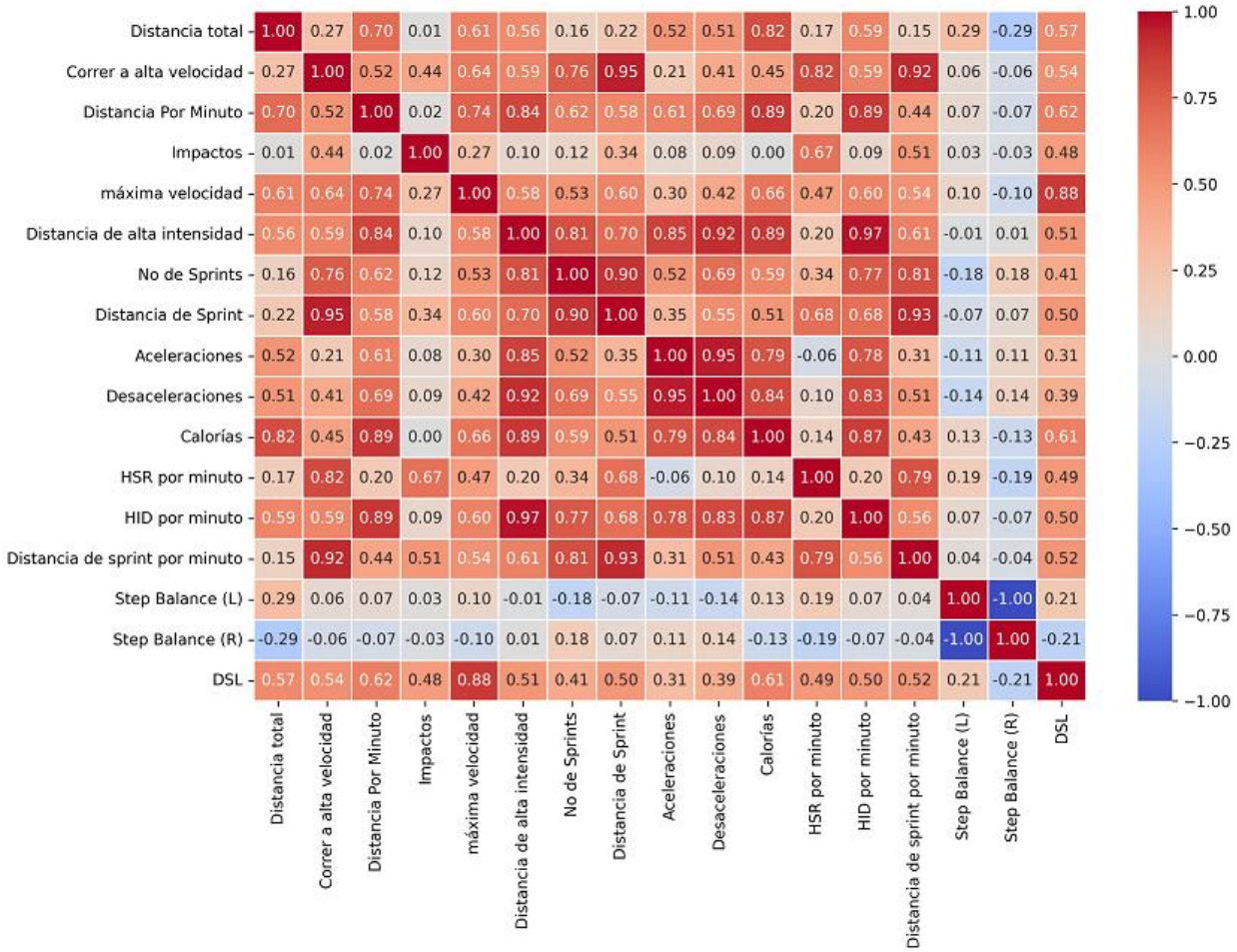
```
Coeficiente de correlación de Pearson con DSL:
DistTotal - DSL: 0.765407
CorrerAltVel - DSL: 0.547009
DistPorMin - DSL: 0.229803
Impactos - DSL: 0.860209
MaxVelocidad - DSL: 0.00164031
DistanciaAltaVelocidad - DSL: 0.856593
NumSprints - DSL: 0.64298
DistSprints - DSL: 0.488224
Aceleraciones - DSL: 0.771975
Desaceleraciones - DSL: 0.858807
```

Resultado Pearson datos Daniel uniendo partidos Fútbol 11 y Fútbol sala

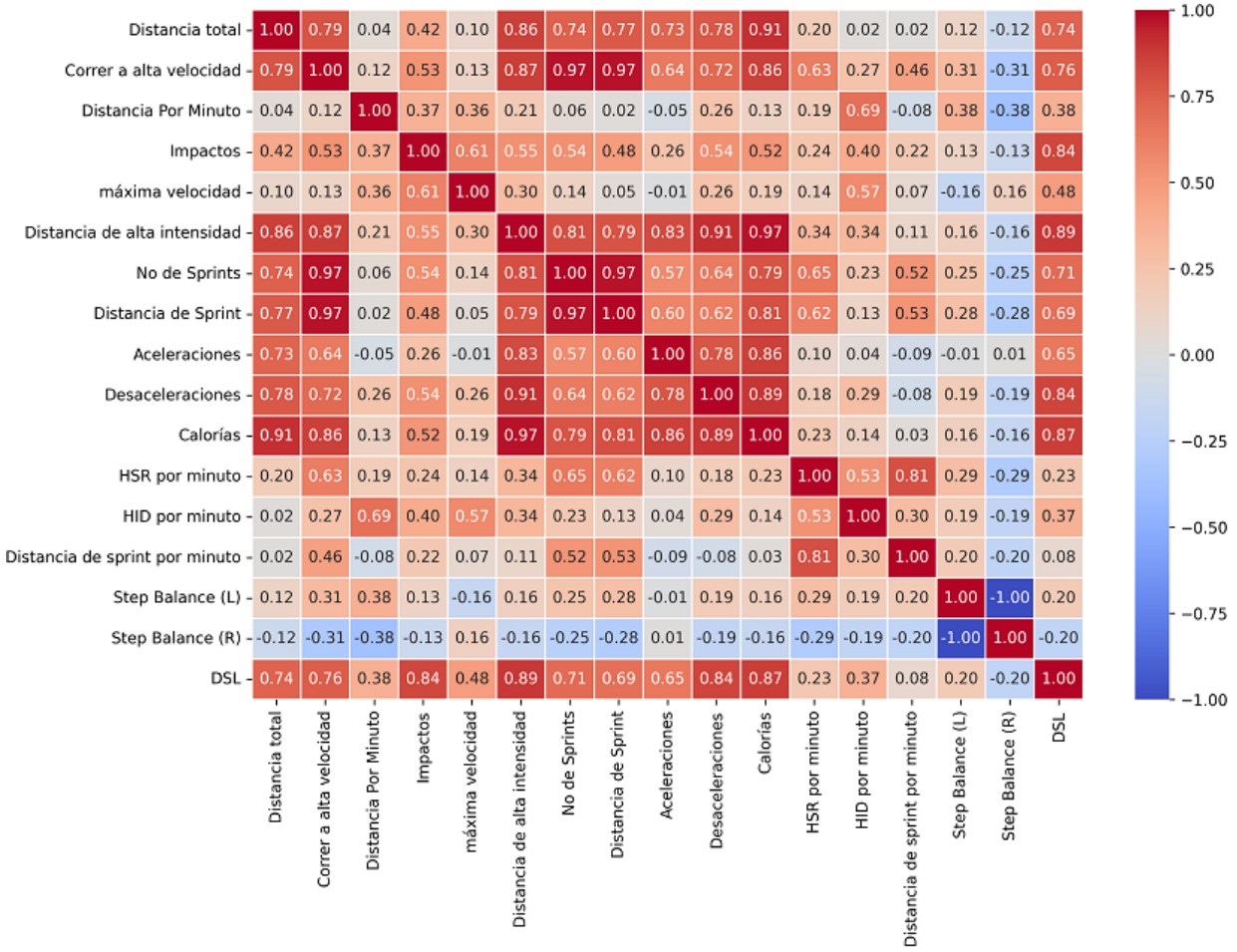
```
Coeficiente de correlación de Pearson con DSL:
DistTotal - DSL: 0.881787
CorrerAltVel - DSL: 0.806061
DistPorMin - DSL: 0.868693
Impactos - DSL: 0.872733
MaxVelocidad - DSL: 0.743778
DistanciaAltaVelocidad - DSL: 0.900243
NumSprints - DSL: 0.769994
DistSprints - DSL: 0.755632
Aceleraciones - DSL: 0.29463
Desaceleraciones - DSL: 0.548507
```

Apéndice E – Matrices correlación

Matrices correlación datos Alejandro entrenamientos Fútbol Sala



Matrices correlación datos Daniel partidos Fútbol Once



Matrices correlación datos Daniel partidos Fútbol Sala