ULPGC
Universidad de
Las Palmas de
Gran Canaria

ESCUELA DE INGENIERÍA
INFORMÁTICA

Grado en Ingeniería Informática

Trabajo Fin de Grado

Software full-stack para la estimación de tiempos de estancia de vehículos pesados en puertos mediante técnicas de aprendizaje automático

Jonay Moreno Sánchez

Tutores

Javier Sánchez Pérez

Christopher Expósito Izquierdo

Las Palmas de Gran Canaria
6 de junio de 2025

ULPGC Universidad de Las Palmas de Gran Canaria | EII ESCUELA DE INGENIERÍA INFORMÁTICA

TFT04

SOLICITUD DE DEFENSA DE TRABAJO DE FIN DE TÍTULO

D./Dª **Jonay Moreno Sánchez**, autor del trabajo **Software full-stack para la estimación de tiempos de estancia de vehículos pesados en puertos mediante técnicas de aprendizaje automático**, correspondiente a la titulación **Grado en Ingeniería Informática**.

SOLICITA

que se inicie el procedimiento de defensa del mismo, para lo que se adjunta la documentación requerida, haciendo constar que

[X] se autoriza / [ ] no se autoriza la grabación en audio de la exposición y turno de preguntas.

Asimismo, con respecto al registro de la propiedad intelectual/industrial del TFT, declara que:
[X] Se ha iniciado o hay intención de iniciarlo (defensa no pública).
[ ] No está previsto.

Y para que así conste firma la presente. (fecha en firma electrónica)

El/La estudiante

Fdo. Jonay Moreno Sánchez

A rellenar y firmar **obligatoriamente** por el/la/los/las tutores

En relación con la presente solicitud, se informa:(firmar donde corresponda)

Positivamente (en caso de detección de copia, esta firma quedará invalidada)	Negativamente (justificación en TFT05)
Fdo. Javier Sánchez Pérez, Christopher Expósito Izquierdo	Fdo. Javier Sánchez Pérez, Christopher Expósito Izquierdo

DIRECTORA DE LA ESCUELA DE INGENIERÍA INFORMÁTICA

Software full-stack para la estimación de tiempos de estancia de vehículos pesados en puertos mediante técnicas de aprendizaje automático

Trabajo Fin de Grado

Jonay Moreno Sánchez

Tutores

Javier Sánchez Pérez

Christopher Expósito Izquierdo

Grado en Ingeniería Informática

Escuela de Ingeniería Informática

Universidad de Las Palmas de Gran Canaria

Las Palmas de Gran Canaria

6 de junio de 2025

Índice general

Índice de figuras

Índice de tablas

Agradecimientos

Quiero darle las gracias a todos los involucrados en este proyecto, ya sea de manera directa como indirecta:

A mi familia, en especial a mi madre, pues siempre ha estado ahí para mí. A mi amigo Pedro Romero, con el que siempre he podido discutir sobre temas informáticos, y al resto de gente que me ha apoyado a lo largo de mis estudios.

A todos los profesores que me han ayudado a formarme desde el colegio hasta la universidad, ya sea como persona o como ingeniero informático.

También agradezco el apoyo de Kaizten Analytics SL. En especial, este trabajo no estaría aquí de no ser por dos magníficos tutores, Javier Sánchez Pérez y Christopher Expósito Izquierdo, que me han guiado a aprender lo máximo posible.

Resumen

La congestión en puertos marinos genera ineficiencias operativas, perdidas económicas en costes logísticos y afecta negativamente al medio ambiente. Esta problemática se debe, en parte, a la falta de sistemas predictivos que faciliten la planificación tanto para transportistas que busquen realizar encargos como para los gestores portuarios encargados de coordinar las operaciones.

Este Trabajo de Fin de Grado propone el desarrollo de una solución software full-stack que ayude a mitigar la congestión portuaria mediante herramientas predictivas que utilicen técnicas de aprendizaje automático. El software propuesto busca mejorar la eficiencia en la gestión del flujo de tráfico y optimizar la asignación de recursos, reduciendo tiempos de espera y beneficiando a los actores involucrados en la cadena logística portuaria.

Abstract

Congestion in seaports generate operational inefficiencies, economic losses on logistic costs, and have a negative impact on the environment. This problematic is partially caused by the absence of predictive systems that facilitate planning both, for transporters looking to make deliveries and for port managers responsible for coordinating operations.

This bachelor thesis aims to develop a full-stack software solution that helps mitigate port congestion through the implementation of predictive tools using machine learning techniques. The goal of the software is to improve the efficiency of traffic flow management and optimize resource allocation, thereby reducing wait times and benefiting the actors involved in the port logistics chain.

Capítulo 1

Introducción

El sector portuario enfrenta de manera recurrente, problemas de congestión generados por el elevado flujo de vehículos pesados, en particular camiones comerciales. Esta situación conlleva retrasos operativos, incrementa los riesgos para la seguridad y ocasiona pérdidas económicas. Asimismo, la congestión vehicular contribuye negativamente al medio ambiente, ya que los vehículos detenidos o en movimiento lento continúan emitiendo gases contaminantes, lo que deteriora la calidad del aire en áreas urbanas próximas.

Las causas de esta congestión son diversas e incluyen una planificación deficiente de las operaciones, la falta de coordinación entre los distintos actores que intervienen en el entorno portuario, la insuficiencia de infraestructuras adecuadas y la carencia de herramientas tecnológicas que faciliten una gestión eficiente del tráfico terrestre. La combinación de estos factores compromete la fluidez operativa del puerto y agrava la problemática existente.

Actualmente, la mayoría de los puertos marítimos no disponen de sistemas informáticos capaces de predecir con precisión el tiempo de permanencia de los vehículos pesados dentro del recinto. La ausencia de estas herramientas predictivas impide que tanto los gestores portuarios como los conductores cuenten con información fiable para planificar adecuadamente sus operaciones.

En el caso específico del Puerto de Santander, que constituye el objeto de estudio del presente trabajo, el sistema de control de accesos se basa en un conjunto de puertas que registran de forma automática las entradas y salidas de vehículos al recinto portuario. Cada evento de acceso queda registrado con una marca temporal y con la identificación de la puerta correspondiente, pero sin un identificador único que permita vincular automáticamente la entrada y la salida de un mismo vehículo. Aunque los registros contienen información temporal, no incluyen detalles adicionales como el motivo del acceso o la tipología del vehículo, lo cual limita su aplicabilidad para ciertos análisis. La figura 1.1 muestra una terminal de accesos que agrupa varias puertas pertenecientes a la zona de Raos, una de las areas de las que se obtuvieron los datos para este trabajo.

El Puerto de Santander, cuya infraestructura general se representa en las figuras 1.2 y 1.3 , se encuentra dividido en distintas zonas destinadas a organizar el flujo de entrada y salida de vehículos. La zona de Raos, ubicada en el extremo sur del puerto, constituye una de las principales áreas operativas y se utiliza para una amplia variedad de actividades logísticas, incluyendo carga y descarga de mercancías, operaciones con grúas, almacenamiento y tránsito de vehículos. Al norte de Raos se sitúa la zona de Maliaño, que alberga operaciones de menor escala. Por su parte, la zona Antonio López Norte, localizada en el paseo de Antonio López en el área urbana de la ciudad, se destina a actividades con

un carácter menos comercial. La identificación y el emparejamiento adecuados de los registros entre estas zonas resultan fundamentales para la estimación precisa del tiempo de estancia de los vehículos en el recinto portuario.

Figura 1.1: Terminal de acceso Raos del Puerto de Santander[1].

La operativa portuaria se ve especialmente comprometida durante los periodos diarios de mayor afluencia, en los cuales la ausencia de estimaciones precisas sobre los tiempos de estancia de los vehículos puede provocar acumulaciones en zonas críticas, demoras en las operaciones y dificultades en la asignación eficiente de recursos, tales como personal o áreas de espera. Esta falta de visibilidad repercute directamente en la eficiencia global de las operaciones, afectando, a su vez, a la experiencia de los transportistas que acceden al recinto.

Con el propósito de mejorar la gestión del tráfico terrestre en el entorno portuario, se propone el desarrollo de una herramienta software basada en técnicas de aprendizaje automático o *machine learning* (ML). Dicha herramienta tiene como objetivo anticipar posibles situaciones de saturación, favoreciendo una gestión proactiva frente a un enfoque reactivo.

Para llevar a cabo esta propuesta, se ha desarrollado una aplicación *full-stack* orientada a los perfiles operativos del entorno portuario. Esta aplicación permite tanto la gestión de reservas de acceso como la visualización de información histórica y de los resultados generados por modelos predictivos. La arquitectura de la solución ha sido diseñada con un enfoque modular, permitiendo la sustitución sencilla de componentes externos como bases de datos, sistemas de mensajería o mecanismos de autenticación.

Asimismo, se han entrenado y evaluado modelos de aprendizaje automático con el objetivo de estimar la duración de la estancia de los vehículos en el recinto portuario. Estos modelos, alimentados con datos históricos reales del Puerto de Santander, permiten

[1]Fuente: `https://kuula.co` de un tour con dron del Puerto de Santander

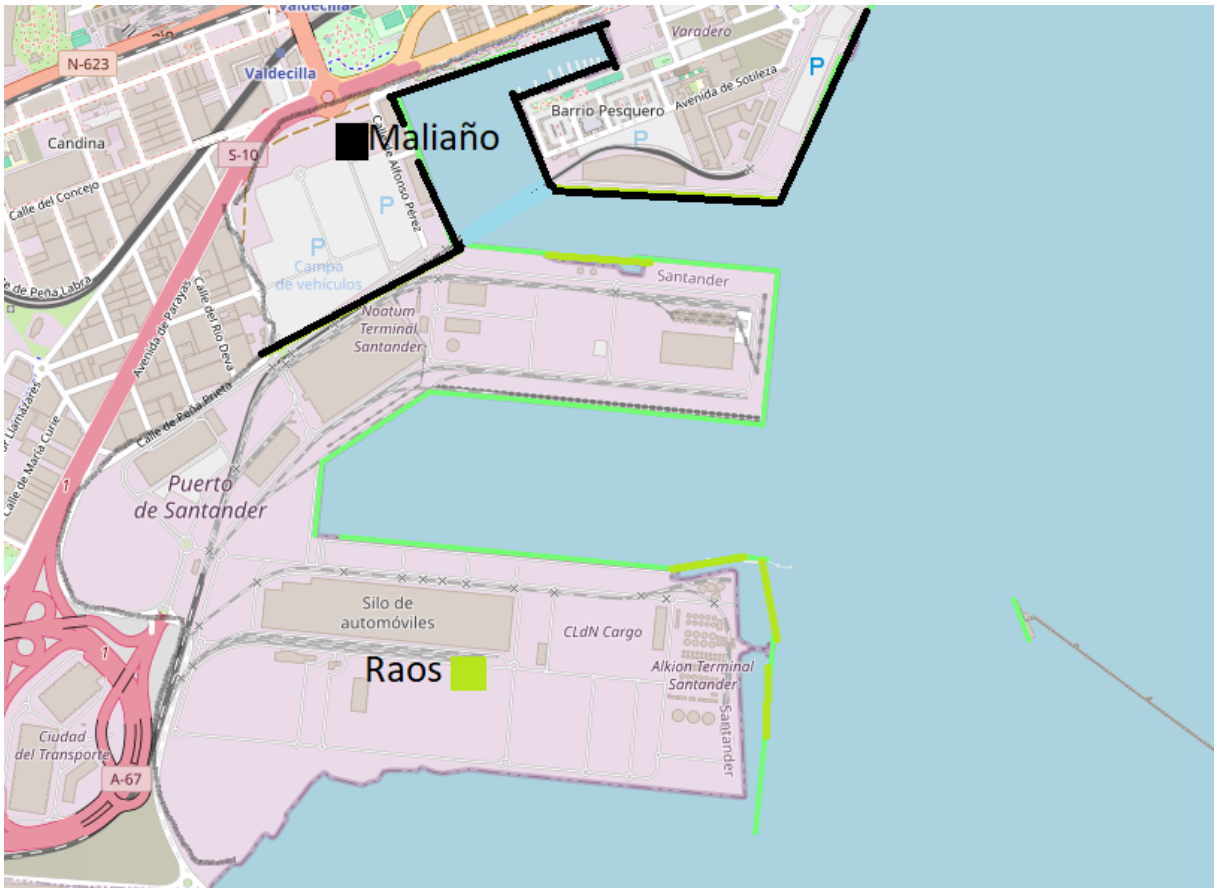

Figura 1.2: Mapa del sur del Puerto de Santander[2].

anticipar periodos de mayor carga operativa y facilitan la planificación estratégica. Cabe señalar que los datos utilizados presentaban una estructura compleja y heterogénea, lo que requirió un proceso significativo de preparación y depuración previo a su uso con fines predictivos. Este desafío relacionado con la calidad y el tratamiento de los datos constituye uno de los principales retos técnicos del proyecto, abordado en detalle en capítulos posteriores.

El trabajo se estructura en dos bloques complementarios. El primero corresponde al desarrollo de la aplicación *full-stack*, orientada a brindar soporte operativo en entornos portuarios mediante funcionalidades de gestión y visualización del acceso de vehículos. El segundo bloque se centra en el diseño, entrenamiento y evaluación de modelos predictivos basados en técnicas de ML, cuyo objetivo es anticipar patrones de ocupación y estancia a partir de datos históricos. Esta separación estructural permite analizar y modelar los datos del Puerto de Santander sin que la aplicación quede limitada exclusivamente a dicho contexto.

Los resultados obtenidos han permitido integrar modelos predictivos con una precisión satisfactoria en una aplicación plenamente funcional. Estos modelos pueden ser fácilmente sustituidos por otros entrenados con conjuntos de datos más recientes —por ejemplo, registros hasta el año 2025—, lo que garantiza la vigencia de la solución en el tiempo y su capacidad de adaptación a la evolución del entorno operativo sin necesidad de rediseñar el sistema completo.

El presente trabajo se dirige tanto a profesionales del ámbito informático interesados

[2]Fuente: OpenStreetMap de `https://www.puertosantander.es/es/muelles` y editado por el autor

en la integración de modelos de ML en aplicaciones logísticas, como a personal técnico portuario que desee explorar las posibilidades que ofrecen las herramientas digitales para la mejora de procesos operativos.

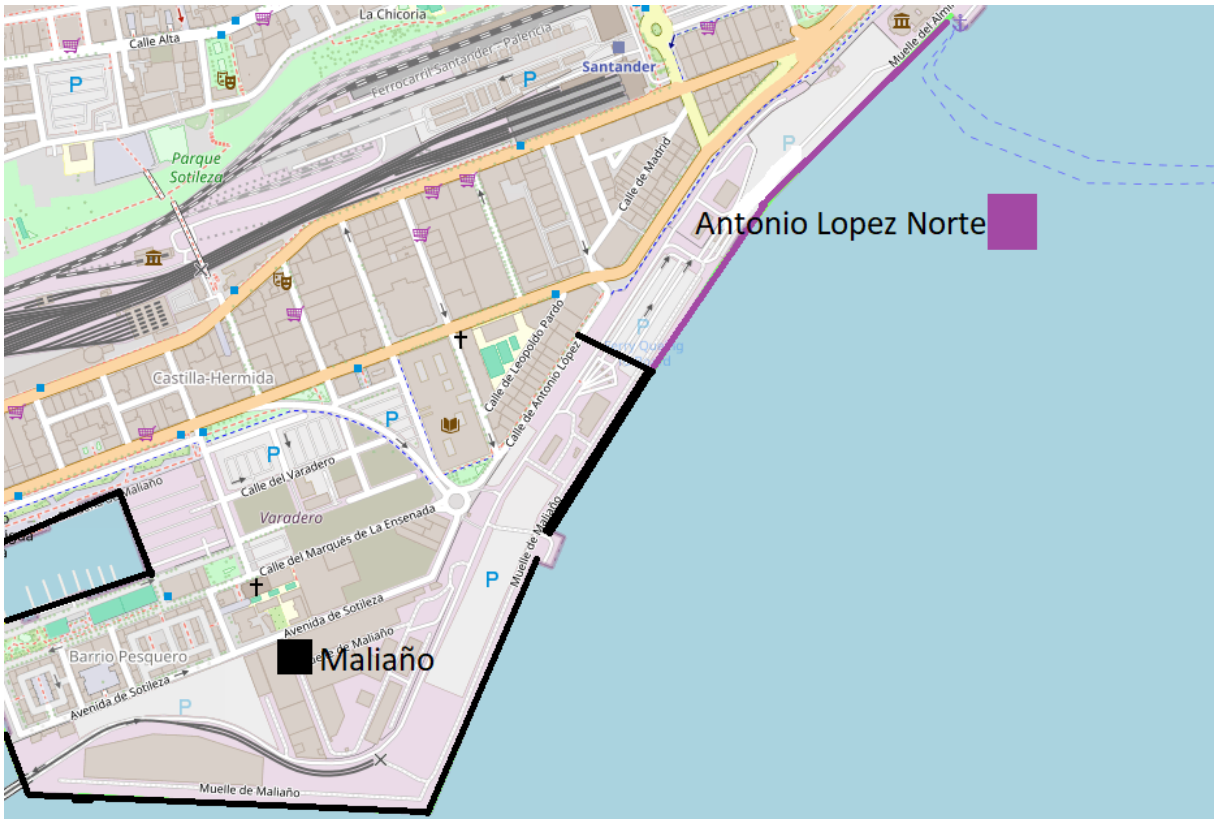

Figura 1.3: Mapa del norte del Puerto de Santander[3].

1.1. Motivaciones

La principal motivación de este Trabajo de Fin de Grado radica en el desarrollo de una solución tecnológica orientada a optimizar la planificación y la toma de decisiones en entornos portuarios, con el objetivo de mitigar los efectos negativos derivados de la congestión terrestre. La implementación de una herramienta predictiva, basada en datos históricos, permite mejorar la asignación de recursos, reducir los tiempos de espera y minimizar el impacto ambiental.

El presente trabajo aborda un problema real identificado en el ámbito profesional, lo que respalda su aplicabilidad y relevancia práctica. Además, se analiza la viabilidad de incorporar modelos de aprendizaje automático en sistemas existentes o en desarrollo, favoreciendo así la digitalización del ecosistema portuario.

1.2. Objetivos

En la propuesta inicial, el objetivo principal consistía en diseñar e implementar modelos de machine learning capaces de predecir con precisión el tiempo de estancia de vehículos pesados en puertos marítimos.

[3]Fuente: OpenStreetMap de `https://www.openstreetmap.org` y editado por el autor

- Identificar y analizar los factores clave que afectan al tiempo de estancia de los vehículos.
- Construir un conjunto de datos adecuado mediante la recopilación, limpieza y transformación de datos provenientes de fuentes portuarias.
- Comparar diferentes técnicas de machine learning (como regresión, árboles de decisión, redes neuronales) para seleccionar la más adecuada.
- Validar los modelos propuestos mediante métricas estándar como RMSE, MAE y R^2.
- Proponer recomendaciones basadas en los resultados para optimizar los procesos logísticos portuarios.

Adicionalmente, aunque no se incluyó explícitamente entre los objetivos, en la metodología propuesta ya se contemplaba la integración de los modelos predictivos en un software desarrollado previamente.

No obstante, a lo largo del desarrollo el alcance del proyecto se fue ajustando para responder a necesidades técnicas y operativas no previstas inicialmente. En particular, se hizo necesario adaptar y extender de forma significativa la herramienta preexistente, dotándola de nuevas funcionalidades orientadas a la gestión de accesos, la visualización de datos históricos y la explotación de resultados generados por los modelos entrenados. Esta adaptación implicó un esfuerzo considerable de desarrollo, lo que llevó a que el componente software adquiriera un protagonismo mayor del inicialmente previsto.

Asimismo, el enfoque del modelado también evolucionó. Aparte de estimar el tiempo de estancia de los vehículos, se incorporó la predicción del volumen de tráfico por franjas horarias, ampliando así el valor añadido de los resultados para la planificación operativa. El objetivo principal de este trabajo es el desarrollo de un sistema de predicción capaz de estimar, con antelación, el tiempo de estancia de los camiones en el entorno portuario, con el fin de facilitar la toma de decisiones logísticas tanto para los gestores portuarios como para los conductores de camiones. Los objetivos secundarios incluyen:

- Analizar e identificar los factores clave que influyen en los tiempos de estancia de los vehículos pesados en el puerto, tales como el tipo de camión, la hora de entrada o la puerta de acceso utilizada.
- Realizar un proceso de limpieza y preprocesamiento de los datos históricos del entorno portuario, garantizando su uniformidad, calidad y adecuación para el entrenamiento de modelos predictivos.
- Evaluar diversas técnicas de aprendizaje automático como regresión lineal y modelos basados en árboles con el propósito de identificar la opción que ofrezca un mejor equilibrio entre precisión, interpretabilidad y tiempo de respuesta.
- Desarrollar una aplicación web que integre los modelos entrenados y permita su uso por parte de usuarios finales, sin requerir conocimientos técnicos sobre su funcionamiento interno.
- Incorporar herramientas de visualización, como gráficos y estadísticas, que presenten la información relevante de forma clara y accesible para los diferentes tipos de usuarios.

1.3. Organización del documento

El contenido del presente documento se organiza en seis capítulos principales, además de varios apéndices complementarios. A continuación, se ofrece una descripción resumida de cada sección:

- *Capítulo 2. Estado del arte:* Se expone una revisión de los trabajos y soluciones existentes en la predicción del tráfico terrestre en puertos mediante técnicas de aprendizaje automático. Asimismo, se analizan estrategias de optimización del flujo vehicular y aplicaciones logísticas relevantes en el contexto portuario.

- *Capítulo 3. Metodología y planificación del trabajo:* Se describen las metodologías empleadas en las distintas fases del proyecto, incluyendo la utilización del modelo CRISP-DM como guía para la construcción de modelos predictivos y el enfoque de desarrollo iterativo incremental para la implementación de la aplicación *full-stack*. Se detalla además la planificación inicial del proyecto y su comparación con la ejecución real, así como las herramientas principales utilizadas durante el desarrollo.

- *Capítulo 4. Desarrollo de la aplicación:* Se documentan los procesos de desarrollo de la aplicación siguiendo una estructura basada en análisis de requisitos, diseño e implementación. La sección de requisitos identifica las funcionalidades y tipos de usuarios, así como los casos de uso y el dominio relacional de las entidades principales; el diseño describe la arquitectura y estructura de la aplicación orientado en capas; y la implementación aborda los aspectos técnicos más relevantes.

- *Capítulo 5. Conjunto de datos:* Se presenta el proceso de obtención, análisis y preprocesamiento de los datos. El propósito de estos pasos es dar lugar a unos datos que puedan ser utilizados para entrenar los modelos predictivos elegidos. El objetivo es resolver dos problemas principales: la estimación del tiempo de estancia de los vehículos en el puerto y la predicción del volumen de tráfico por franjas horarias. Finalmente, se seleccionan los modelos más adecuados para su integración en la aplicación desarrollada.

- *Capítulo 6. Modelos de predicción y resultados*: Se presentan las métricas utilizadas para evaluar la calidad de los modelos predictivos, como el error absoluto medio (MAE), el error cuadrático medio (RMSE) y el coeficiente de determinación (R^2). A continuación, se describen los modelos desarrollados para resolver los dos problemas presentados en el capítulo 5. Finalmente, se analizan los resultados obtenidos, se comparan los desempeños de los modelos y se seleccionan los más adecuados para su integración en la aplicación desarrollada.

- *Capítulo 7. Conclusiones y trabajo futuro:* Se sintetizan los principales logros alcanzados, se discuten las limitaciones identificadas y se proponen posibles líneas de mejora y desarrollo futuro. Así como a los avances correspondientes al desarrollo del autor.

- *Apéndices*. Se recogen las competencias adquiridas durante el proyecto, la contribución a los Objetivos de Desarrollo Sostenible (ODS) y un manual de usuario que incluye la interfaz (vistas) de la aplicación desarrollada.

Capítulo 2

Estado del arte

La creciente complejidad de las operaciones portuarias [1], junto con el aumento sostenido en los volúmenes de carga, ha impulsado la búsqueda de soluciones innovadoras para gestionar el tráfico terrestre en entornos portuarios de manera más eficiente. En particular, la gestión de camiones y otros vehículos pesados ha cobrado especial relevancia debido a su impacto directo en la congestión portuaria, los tiempos de espera y la eficiencia operativa general.

En este contexto, las tecnologías modernas orientadas a la digitalización se ven apoyadas por iniciativas, como la de Puertos 4.0 [2] y los programas de Puertos Inteligentes [3]. Estas iniciativas, han comenzado a incorporar técnicas avanzadas de análisis de datos y ML, promoviendo la adopción de tecnologías como el *big data*, el Internet de las Cosas y el ML para mejorar la eficiencia, sostenibilidad y seguridad de los entornos portuarios.

En particular, Puertos 4.0 [2] es impulsado por Puertos del Estado que financia el desarrollo de soluciones digitales en la operativa y logística portuaria. A nivel global, este se alinea con la tendencia de convertir las infraestructuras tradicionales en infraestructura inteligente, capaz de responder a las demandas del comercio marítimo mediante automatización y conectividad [3].

Por otro lado, los Puertos Inteligentes [1] son una evolución estratégica del modelo portuario, orientados a integrar tecnologías avanzadas en todos los niveles de operación portuaria. En este sentido, los Puertos Inteligentes se han consolidado como una respuesta global a los retos derivados del aumento del comercio marítimo y la necesidad de puertos más eficientes, seguros y duraderos.

Este capítulo presenta un desglose del estado del arte en dos dimensiones clave: por un lado, la predicción del comportamiento del tráfico terrestre en puertos mediante ML; y por otro, la optimización del tráfico interno portuario, abarcando sistemas de gestión y diversas estrategias para mejorar la operativa.

2.1. Predicción mediante técnicas de ML en puertos

Para abordar o intentar resolver el problema de congestión portuaria, se han realizado análisis estadísticos básicos o hecho uso de la experiencia de un empleado a cargo que se haya aprendido las tendencias del puerto. No obstante, estas aproximaciones resultan limitadas para la multitud de factores que influyen en este tiempo, algunos de los cuales son: el volumen de camiones en tránsito, nivel de actividad de las grúas, hora del día, tipo de carga, trámites de aduana, etc. En los últimos años [4], han cobrado protagonismo las técnicas de ML que buscan dar soluciones consistentes a los problemas logísticos de este

tipo, dado que tienen la capacidad de aprender patrones y relaciones complejas a partir de datos históricos.

Trabajos académicos han demostrado el potencial del ML para predecir con mayor precisión los tiempos de estancia de contenedores y camiones en puertos. Se entrenó un modelo que utiliza redes neuronales para predecir el tiempo de permanencia de contenedores de importación en una terminal portuaria [5]. Este modelo neuronal incorporó múltiples variables como: fecha de llegada, origen, tipo, tamaño del contenedor, etc. Logrando el resultado de estimaciones con precisión de la estancia de hasta un 65 %.

En un estudio [6], se aplicaron métodos de ML avanzados para modelar el volumen diario de camiones que frecuentan las terminales en el Puerto de Houston. Se compararon algoritmos como las máquinas de vectores de soporte (Support Vector Machines, SVM) y los procesos gaussianos (Gaussian Processes) frente soluciones anteriores basadas en redes neuronales tradicionales de múltiples capas con propagación hacia adelante (*multilayer feedforward neural networks*). Los resultados evidenciaron que los modelos basados en SVM y procesos gaussianos ofrecían mejores predicciones y requerían menos ajustes, lo que promueve el uso de técnicas sofisticadas de ML en este campo.

Las SVM son algoritmos supervisados utilizados en problemas de clasificación y regresión [7]. Su objetivo es encontrar la mejor forma de separar los datos en distintos grupos, trazando una frontera que maximice la separación entre las clases. Además, permiten modelar relaciones complejas o no lineales mediante funciones núcleo (*kernel*), lo que las hace buena elección para una amplia variedad de problemas reales.

Con el auge de la tendencia conocida como *big data* en la industria portuaria, estudios han explorado técnicas de aprendizaje profundo para mejorar las predicciones a corto plazo del flujo de camiones. Un modelo neuronal de propagación hacia adelante *feed-forward* fué propuesto con el objetivo de predecir el tráfico saliente de camiones con una hora de antelación en el puerto de Rotterdam [8]. Para esto se incorporan datos compartidos en tiempo real entre actores logísticos como terminales y flujos de los conductores de camiones. Este enfoque demostró poder anticipar exitosamente el tráfico saliente de camiones.

Además de estudios académicos, el uso de ML para estimar tiempos de estancia ha empezado a aparecer en proyectos industriales. Uno de ellos es el del puerto de Hamburgo [9], en donde se implantó en 2020 un módulo de inteligencia artificial con el objetivo de procesar datos en tiempo real para predecir los tiempos de recogida de contenedores. En el sistema de control de una de sus terminales de contenedores, esta solución desarrollada junto a consultores externos se integró en el software de gestión del entorno portuario, permitiendo optimizar la secuencia de movimientos necesarios para la gestión de los contenedores según la hora prevista de llegada de los camiones. De esta forma, si un contenedor fuera a quedarse en la zona de espera durante más tiempo porque su camión llega tarde, el sistema podrá retrasar la manipulación de este contenedor y priorizar otras tareas más urgentes, incrementando la productividad del almacén.

Tras la implementación de la herramienta, se observó una mejora en la productividad del entorno, con una reducción significativa de los tiempos de espera de los camiones y un aumento en la eficiencia de manipulación de contenedores.

España también ha impulsado iniciativas de Puertos 4.0 en el tiempo de estancia. Cabe citar un proyecto piloto en el Puerto de Santander [10], el cual se centró en el análisis avanzado y predictivo de la estancia y tránsito de vehículos pesados en terminales portuarias. Mediante técnicas de inteligencia artificial, este proyecto resultó en un producto del tipo prueba de concepto. Este piloto sienta las bases de una herramienta de apoyo a

la operativa diaria, al predecir llegadas, tiempo de estancia y alertar con antelación de posibles picos de congestión que permitan a la autoridad portuaria tomar medidas para mitigar dicha congestión.

2.2. Optimización interna del tráfico portuario

Una estrategia utilizada para regular la congestión portuaria es el uso del sistema de citas previas de camiones (*truck appointment systems*, TAS) [11]. En un TAS, cada camión recibe un tiempo asignado para entrar al puerto y realizar las operaciones y trámites necesarios, realizando así una distribución de la llegada de vehículos en donde no se produzca congestión portuaria en momentos pico del día. Bastantes puertos, especialmente aquellos centrados en el transporte de mercancía han adoptado el sistema de cita TAS, y estudios académicos desvelan los beneficios de este sistema. Según una publicación [12], los TAS han sido considerados elementos clave para evitar la saturación y congestión portuaria, puesto que reducen tanto periodos de espera como mejoran la operativa general de las puertas de acceso al recinto.

El problema comentado de la optimización de las citas de camiones ha sido objeto de la investigación operativa. En un estudio [13], la asignación de citas fue optimizada tratando al problema matemáticamente como si se de uno de programación entera, buscando minimizar el tiempo de los camiones en la terminal de una puerta de acceso. La solución propuesta de la investigación, fue probada en escenarios simulados y demostró que una asignación óptima de las franjas horarias reducira en gran medida las esperas ubicadas en las puertas de acceso y disminuira el tiempo de estancia total de los camiones.

Otros trabajos han investigado la problemática de congestión mediante algoritmos metaheurísticos [14] para intentar resolver las variaciones del problema de citas cuando la cantidad de camiones es alta o existe cierta incertidumbre sobre la llegada de estos camiones. En los casos estudiados, el objetivo es distribuir las llegadas al recinto de manera que no se produzca una acumulación de vehículos, lo que produce congestión y genera problemas para todos los actores involucrados.

En general, la evidencia presentada [13] sugiere que una adecuada optimización en la asignación de citas puede lograr un equilibrio eficiente entre la llegada de camiones y la capacidad operativa del puerto. Esta sincronización no solo mejora el rendimiento portuario, sino que también genera beneficios colaterales en otros sectores logísticos, como la mejor coordinación entre buques y trenes, reduciendo así los efectos de retrasos en cascada.

Por otra parte, la gestión inteligente del tráfico interno portuario mediante técnicas de análisis de datos y visión por computador en tiempo real constituye un área de investigación en expansión. Esta línea implica la recolección de información relevante a través de sensores, cámaras y otros dispositivos, que permiten anticipar niveles de congestión y planificar de forma dinámica el estacionamiento [15][9].

En este contexto, los sistemas de transporte inteligente integran tecnologías de control, comunicación y procesamiento para mejorar tanto la eficiencia como la seguridad de los vehículos. Según un informe [16], el rastreo en tiempo real de la carga almacenada en los puertos permite aplicar esquemas logísticos como el justo a tiempo (*just in time*, JIT), una estrategia de almacenamiento, basada en recibir y entregar mercancia justo cuando se necesita, reduciendo desplazamientos innecesarios y minimizando los tiempos de espera en rutas y accesos portuarios. Estos sistemas contribuyen a disminuir la congestión y a mejorar la seguridad, al tiempo que fortalecen la coordinación entre los distintos actores

del ecosistema portuario.

Un enfoque contemporáneo para optimizar la circulación en entornos portuarios se basa en técnicas de ML, en particular mediante algoritmos de aprendizaje por refuerzo implementados en simulaciones [17]. Estos modelos son capaces de aprender políticas óptimas a partir de la interacción con un entorno simulado, recibiendo recompensas por decisiones acertadas y optimizando métricas de rendimiento acumulado. Este enfoque simula una terminal de contenedores, en la cual un algoritmo de aprendizaje por refuerzo profundo (Deep Double Q-Network) gestiona dinámicamente la asignación de camiones a diversas tareas de transporte dentro del puerto. El objetivo del modelo es minimizar el tiempo máximo de finalización de las operaciones portuarias (grúas de muelle, grúas de patio, etc.), coordinando eficientemente la disponibilidad de camiones. Los resultados obtenidos muestran que los patrones aprendidos por el modelo reducen significativamente los tiempos de espera de las grúas y mejoran la utilización de los camiones en comparación con métodos heurísticos tradicionales. Este trabajo ilustra el potencial de la inteligencia artificial para descubrir estrategias de gestión más eficaces que aquellas diseñadas manualmente, especialmente en sistemas complejos con múltiples entidades interactivas.

Naturalmente, la optimización del tráfico interno portuario también abarca otros aspectos interrelacionados. Una vez que los camiones ingresan al recinto portuario, su tiempo de permanencia hasta la salida está determinado por una secuencia de sub-operaciones (inspecciones, carga y descarga, tránsito entre patios, entre otras). La sincronización entre estas actividades resulta crítica para el rendimiento global del sistema. Estudios basados en simulación de operaciones portuarias han permitido evaluar cómo distintas configuraciones de rutas internas y prioridades en colas afectan el desempeño del sistema [18]. Asimismo, se han investigado problemas asociados a la ruta óptima que deben seguir los camiones dentro del puerto para movilizar contenedores entre el muelle y el patio. Estos algoritmos de ruteo buscan minimizar la distancia recorrida y el tiempo de transporte, considerando siempre restricciones operativas relevantes como la no interferencia con otros vehículos, la no generación de retrasos en buques y otras limitaciones del entorno portuario.

2.3. Aplicaciones en logística y transporte portuario

El uso de técnicas de ML no se limita a la predicción del tiempo de estancia o del tráfico portuario, sino que abarca un espectro mucho más amplio de aplicaciones en logística y transporte marítimo. En particular, la denominada logística 4.0 se caracteriza por la utilización intensiva de datos y algoritmos inteligentes para optimizar cadenas de suministro completas. A continuación, se presentan algunas de las aplicaciones más relevantes como contexto general del problema específico abordado en este trabajo.

En los últimos años, un estudio [18] ha revelado tendencias en la aplicación de ML en entornos portuarios. Se observa un creciente interés en el desarrollo de modelos predictivos y prescriptivos, aunque persiste una escasez de investigaciones que integren estas técnicas con métodos clásicos de investigación operativa. Entre las pocas investigaciones identificadas, aquellas que combinan ambas aproximaciones muestran tasas de éxito significativamente superiores. Asimismo, destaca el auge de modelos basados en aprendizaje por refuerzo, que ofrecen mejoras notables frente a enfoques estáticos.

El estudio también subraya el potencial de combinar modelos predictivos y prescriptivos, demostrando cómo el aprendizaje por refuerzo puede resolver problemas complejos de optimización en el entorno portuario. A partir de una revisión sistemática de 124 artículos,

se concluye que la integración de técnicas de ML con métodos de operación puede mejorar sustancialmente la eficiencia, especialmente en áreas como el mantenimiento y el control operativo. Asimismo, se enfatiza la importancia de incorporar las salidas de los modelos predictivos en procesos de toma de decisiones estructurados, orientados a optimizar la operativa portuaria.

Un caso destacado en España es el proyecto del Puerto de Bilbao [19], donde en 2020 se implementó un Agile Recognition Software desarrollado por la empresa AllRead. Este sistema utiliza inteligencia artificial y visión por computador para automatizar los accesos terrestres al recinto portuario. Desde su puesta en marcha, ha ampliado su cobertura de dos a diez carriles, mejorando la eficiencia operativa, la trazabilidad de las mercancías y la seguridad mediante la detección automática de materiales peligrosos. Como resultado, se han reducido tiempos de espera, costes adicionales y emisiones contaminantes, posicionando al Puerto de Bilbao como una infraestructura modernizada y digitalizada. AllRead ha continuado mejorando su tecnología, incorporando recientemente mejoras en el reconocimiento óptico de caracteres adaptado a entornos portuarios [20]. Este tipo de tecnologías ópticas permite una adquisición robusta y digital de datos.

En el ámbito empresarial, destaca el proyecto SiMU-PORT [20], [21], desarrollado por ISOIN en el marco del programa Puertos 4.0. Esta iniciativa se centra en la optimización de operaciones logístico-portuarias vinculadas al tráfico de pasajeros y vehículos, integrando técnicas de simulación de eventos discretos con modelos de ML. El sistema incorpora un proceso ETL (extracción, transformación y carga) que consolida datos provenientes de múltiples sistemas de información portuarios en un almacén centralizado, alimentando modelos predictivos sobre los flujos de personas, vehículos y ferris. Mediante simulación, se evalúan distintos escenarios operativos, optimizando factores como tiempos de espera, cuellos de botella, costes y emisiones. Los resultados de la simulación sirven como entrada a un algoritmo prescriptivo entrenado con diversos escenarios, que genera recomendaciones para la gestión de recursos y planificación operativa, con el objetivo de aumentar la eficiencia y sostenibilidad del entorno portuario.

Es importante subrayar que la transformación digital en los puertos no ocurre de forma aislada, sino como parte integral de la transición hacia la logística 4.0 y los puertos inteligentes. Las soluciones de predicción y optimización del tráfico de camiones, tratadas en las secciones anteriores, conectan eficazmente el flujo marítimo con la logística terrestre. El éxito de estas técnicas depende en gran medida de la colaboración con los transportistas, promoviendo el uso de herramientas digitales accesibles y fáciles de implementar, así como de su integración con la cadena logística externa. Por ello, las iniciativas de puertos inteligentes se alinean con estrategias más amplias como la movilidad inteligente y la logística conectada, buscando una optimización integral del transporte de extremo a extremo. La convergencia tecnológica con el internet de las cosas, el *big data* y los algoritmos de ML continúa revolucionando las operaciones logísticas.

En base a lo anterior, puede afirmarse que el panorama actual muestra un ecosistema prometedor de estudios y proyectos con gran potencial de impacto. Por un lado, se están utilizando modelos predictivos como redes neuronales, regresión o métodos basados en SVM para anticipar tiempos de estancia y otros eventos operativos. Por otro, se aplican herramientas de optimización que aprovechan estas predicciones y datos en tiempo real para mejorar la gestión del tráfico interno y la asignación de recursos. Esta combinación permite una toma de decisiones basada en datos, maximizando el uso de recursos sin requerir ampliaciones físicas.

No obstante, también se identifican desafíos, especialmente en la integración efectiva

de estas técnicas en las operaciones diarias. La colaboración entre los actores implicados, facilitando el intercambio de datos y experiencias, será clave para el desarrollo de herramientas ajustadas a las necesidades de cada puerto. Estos retos representan oportunidades interesantes de investigación y desarrollo, lo que sugiere que la gestión del tráfico portuario seguirá avanzando al ritmo de la evolución tecnológica, sincronizado con las exigencias del mercado.

En el contexto del trabajo, el desarrollo de una solución que integre modelos que utilicen técnicas de ML para estimar de forma precisa los tiempos de estancia de vehículos pesados en puertos representa una línea de trabajo con gran potencial. A pesar de la existencia de multiples tecnologías predictivas y de optimización en puertos, persiste la necesidad de herramientas que se adapten a contextos locales específicos, con integración sencilla en la operativa diaria y capacidad de generalización a partir de datos históricos reales. Este trabajo se alinea con dicha necesidad, proponiendo una aplicación software *full-stack* orientada a proporcionar estimaciones precisas del tiempo de estancia de camiones en recintos portuarios, combinando las técnicas anteriores con un diseño multicapa orientado a la web moderna que permita la visualización y explotación de los resultados por parte de usuarios operativos.

Capítulo 3

Metodología y planificación del trabajo

Este capítulo recoge la planificación y preparación inicial necesarias para el desarrollo del proyecto, así como las herramientas principales empleadas durante el trabajo.

A lo largo de este capítulo se detallan las herramientas utilizadas en cada uno de estos bloques, la planificación temporal establecida al inicio, las tareas principales que se definieron, así como las adaptaciones realizadas durante el desarrollo.

Aunque se tuvo en cuenta la planificación inicial, el desarrollo del proyecto requirió cierta flexibilidad, adaptándose a medida que surgían imprevistos o nuevas necesidades. Este enfoque permitió redistribuir esfuerzos de forma progresiva sin comprometer los objetivos finales.

3.1. Metodología

Como se comentó en el capítulo de Introducción 1, este trabajo se divide en dos grandes partes. Una, de desarrollo software, y otra de desarrollo de modelos utilizando técnicas de ML.

Para abordar estas dos cuestiones se utilizaron dos metodologías distintas, pues el desarrollo de software no se puede hacer con la metodología utilizada de ciencia de datos, planificada inicialmente.

Para el apartado de ML del desarrollo del trabajo, se utilizo la conocida metodología de ciencia de datos, Cross Industry Standard Process for Data Mining [22] o más conocida como CRISP-DM. Esta es una metodología diseñada para ser iterativa y flexible y se divide en unos procedimientos específicos:

1. Entender el negocio
2. Entender los datos
3. Preparar los datos
4. Modelado
5. Evaluación
6. Despliegue

Este trabajo dio comienzo con una fase exploratoria de análisis de los datos disponibles, detectando de esta manera limitaciones en los datos y planteando hipótesis sobre el comportamiento dentro del recinto portuario analizado. A medida que se profundizaba en el análisis, el objetivo específico del modelo cambió: inicialmente centrado en predecir el tiempo de estancia portuaria, se replanteó hacia la estimación del volumen de tráfico.

Durante la fase de preparación de datos se aplicaron técnicas de limpieza, transformación y codificación de variables, adaptando el conjunto de datos a los requisitos de algoritmos de modelado. Posteriormente, se exploraron enfoques predictivos, desde modelos lineales hasta técnicas basadas en árboles. Finalmente, los modelos ya entrenados con mejor rendimiento fueron integrados en el software desarrollado, cerrando así el ciclo de esta parte del desarrollo.

La elección de CRISP-DM se justifica por su flexibilidad, su capacidad de adaptarse a situaciones donde el análisis de datos no es lineal ni cerrado desde el principio. En este trabajo, la comprensión de los datos y sus limitaciones llevó a redefinir el enfoque inicial del modelo, y la estructura de CRISP-DM permitió reorientar el desarrollo sin interrumpir el progreso del proyecto.

La otra metodología utilizada para el proyecto de desarrollo de software *full-stack*, se llevó a cabo mediante un desarrollo iterativo incremental [23]. La metodología aplicada en esta parte combina ciclos de desarrollo iterativos con una construcción basada en incrementos. Permitiendo realizar entregas parciales del resultado final en cada ciclo, incorporando nuevas funcionalidades y mejoras basándose en sugerencias, permitiendo la adaptación a cambios y mejora continua del proyecto.

Si bien el objetivo final era integrar modelos predictivos en la aplicación, los pasos intermedios se definieron de forma evolutiva en función del progreso y las necesidades detectadas.

Se establecieron ciclos de desarrollo cortos (de dos semanas aproximadamente), tras los cuales se establecían nuevos objetivos como la implementación de una funcionalidad específica, integración de datos, y más en reuniones con los tutores.

Estas sesiones de seguimiento permitieron incorporar mejoras funcionales no contempladas en un primer momento. Un ejemplo concreto de estas adaptaciones fue la inclusión de gráficas de resumen sobre los accesos al puerto, que surgieron como una necesidad detectada durante las pruebas internas, y que aportaron valor añadido a la visualización de resultados y la toma de decisiones operativas.

Durante el desarrollo, se realizaron seis iteraciones principales, cada una centrada en objetivos definidos dependiendo del estado del proyecto en ese momento. A continuación, se destacan los hitos de cada iteración:

- **1ª iteración**: Definición del núcleo (dominio) y diseño de la arquitectura general de la aplicación. En esta etapa se sentaron las bases estructurales y herramientas a utilizar.
- **2ª iteración**: Desarrollo de la lógica de negocio (núcleo y adaptadores esenciales) de la aplicación y diseño inicial del *front-end*, centrado en una versión básica de la interfaz.
- **3ª iteración**: Implementación completa de la interfaz de usuario y funcionalidades asociadas a la importación y gestión masiva de datos.
- **4ª iteración**: Desarrollo de un sistema de mayor velocidad para la carga de datos y de una visualización gráfica integrada, con el objetivo de representar información

básica.

- **5ª iteración**: Integración inicial de un modelo de predicción entrenado, validando la conexión entre *back-end* y el componente de inferencia.
- **6ª iteración**: Finalización del sistema con la integración de los modelos definitivos, así como la adaptación del comportamiento de la aplicación en función del rol del usuario.

El enfoque iterativo incremental permitió adaptar el desarrollo *full-stack* a descubrimientos realizados en paralelo con el apartado de *machine learning*, así como a responder con agilidad a cambios o nuevas necesidades.

3.2. Planificación

En esta sección se desglosarán las diferencias que ocurrieron entre lo planificado inicialmente y lo que realmente ha sucedido. Para ello tenemos la tabla conteniendo la planificación inicial: 3.1

Tabla 3.1: Fases y tareas desarrolladas en el proyecto.

Fases	**Horas**	**Tareas**
Comp. Negocio / Comp. Datos	110	Tarea 1.1: Identificar fuentes de datos relevantes
		Tarea 1.2:Gestionar el acceso a los datos, cumpliendo las políticas de privacidad.
		Tarea 1.3: Aprender a utilizar las librerías consideradas.
Prep. Datos / Modelado	110	Tarea 2.1: Limpieza de datos y transformación de variables
		Tarea 2.2: Selección de técnicas y entrenamiento
Evaluación / Despliegue	50	Tarea 3.1: Integración en el software *full-stack* .
		Tarea 3.2: Implementar mejoras que se consideren.
Documentación / Presentación	30	Tarea 4.1: Documentar los resultados obtenidos
		Tarea 4.2: Preparar la presentación del TFT

En un principio se planteó un proyecto centrado en las técnicas de ML, el cual se centraría en el desarrollo de estas técnicas utilizando CRISP-DM como se observa en la Tabla 3.1 por lo que la asignación al apartado software fue pequeña y se pensaba realizar durante la fase de despliegue de la aplicación. Sin embargo, a medida que avanzó el desarrollo, fue evidente que integrar correctamente los modelos en una aplicación funcional y usable requeriría de un esfuerzo considerablemente mayor al estimado.

Desde el inicio del trabajo, la recopilación de los datos se desarrolló de forma más fluida de lo esperado. En la tarea 1.1, se lograron obtener los datos necesarios de manera eficaz debido a que estos ya estaban disponibles y bien organizados en los documentos pertinentes, por lo que no hubo necesidad de construir nuevos conjuntos desde cero realizando procesos complejos de recopilación manual. Esto permitió avanzar rápidamente hacia las fases de análisis y preparación. Esta eficiencia inicial facilitó la comprensión del conjunto de datos y contribuyó a un inicio más rápido del desarrollo.

En cuanto a la tarea 1.2, centrada en la gestión del acceso a los datos y su adecuación legal, también fue más sencilla de lo previsto. Los únicos datos personales contenidos en los registros eran los documentos nacionales de identidad (DNI) de los conductores, lo que simplificó la aplicación de medidas de anonimización. Aunque se planteó eliminar la sección correspondiente a la normativa de protección de datos por considerarse poco relevante, se decidió mantenerla dada la manipulación directa de identificadores personales durante el preprocesamiento.

La tarea 2.1, centrada en la limpieza y transformación de datos, se desarrolló según lo planificado. Aunque se encontraron problemas menores, como lidiar con estructuras poco uniformes y emparejar accesos de entrada y salida al puerto, estos posibles retos ya se habían tenido en cuenta desde la fase de planificación, debido a que es común que conjuntos de datos necesiten de estos procesos. Gracias a la preparación anterior, la tarea se pudo llevar a cabo en el tiempo estimado. Esta preparación afectó directamente a la tarea 2.2, relacionada con la selección de técnicas y entrenamiento de modelos, fue realizada en paralelo con la anterior. El trabajo previo de limpieza permitió disponer de un conjunto de datos bien estructurado, lo que redujo la necesidad de realizar muchos ajustes o pruebas adicionales, haciendo que la tarea se completara en menos tiempo del estimado.

Respecto a la tarea 3.1, centrada en la integración de los modelos predictivos en el software *full-stack*, su ejecución resultó bastante más exigente de lo que se había estimado. No solo implicó conectar la aplicación con los modelos una vez entrenados, sino también asegurar que los resultados fueran accesibles desde el *front-end*. Esta integración exigió iteraciones adicionales y ajustes técnicos tanto en la lógica de negocio como en la visualización.

Además, se implementaron una serie de mejoras adicionales no contempladas en la planificación inicial, pero que pueden considerarse una extensión de la tarea 3.2. Estas mejoras permitieron integrar y reutilizar los datos de forma más eficiente dentro de la propia aplicación, lo que tuvo un impacto positivo durante el análisis posterior, tanto en la visualización como en la validación de los modelos.

En resumen, aunque se respetó el total de horas planificadas, la distribución real del tiempo fue distinta. Las fases relacionadas con la puesta en producción y mejoras funcionales absorbieron más tiempo del esperado, mientras que las fases iniciales fueron completadas antes de lo previsto. Gracias a las metodologias utilizadas, fue posible redistribuir este tiempo sin afectar negativamente a los objetivos generales del trabajo.

3.3. Herramientas utilizadas

Durante el desarrollo de este trabajo se han empleado múltiples herramientas tanto para el desarrollo de la aplicación como para la construcción y evaluación de los modelos predictivos. Esta sección presenta las herramientas principales, separadas según su ámbito de aplicación, junto con una breve descripción de la herramienta y justificación de su uso.

3.3.1. Herramientas para el desarrollo de la aplicación

En primer lugar, para el control de versiones del código fuente ha sido gestionado mediante GitHub. Esta es una plataforma en la que los desarrolladores pueden guardar el histórico del código de proyectos para compartirlo de manera sencilla y poder trabajar en el en paralelo. Entre otros, actuó como mecanismo de recuperación en eventos inesperados.

Se ha elegido utilizar está herramienta debido a la utilidad que se obtiene a través de un histórico riguroso donde se pueden observar los cambios a lo largo del tiempo en la aplicación. Otros factores importantes han sido el respaldo que se ofrece mediante GitHub Actions una herramienta que permite automatizar tareas.

El despliegue fácilitado de la aplicación contenida en el repositorio, se basa en una serie de contenedores Docker, lo que a su vez permite un despliegue consistente sin necesidad de realizar configuraciones extensas, este se ha utilizado en conjunto con Docker Compose. Esta herramienta permitió generar imágenes de contenedores grupales garantizando consistencia entre entornos de desarrollo, sin dar lugar a problemas especificos de la máquina utilizada. Adicionalmente hacer uso de esta tecnologia Docker da pie al uso de Docker Hub, permitiendo el despliegue del proyecto a traves de un simple archivo.

Uno de los contenedores desplegados contiene el sistema de gestión de autenticación de código abierto, Keycloak desarrollado por Red Hat. Proporciona funcionalidades como autenticación basada en roles, gestión de sesiones y más en los protocolos OAuth2, OpenID Connect y SAML 2.0.

En el contexto de este trabajo, Keycloak fue elegido por su facilidad para gestionar distintos roles de usuario, y su compatibilidad con el uso de JWT, un estándar de autenticación. Además, cuenta con una interfaz de administración robusta que permite una configuración completa.

Alternativas similares son Auth0 o Firebase Authentication. Sin embargo, estas son soluciones externas, que implican costes adicionales no justificables para este trabajo. Keycloak, al ser alojado localmente, ofrece control total de configuración y despliegue. Los detalles técnicos de integración de Keycloak se abordan en la sección de implementación del desarrollo 4.

El sistema mensajeria de la aplicación utiliza RabbitMQ en forma de contenedor, permitiendo una comunicación asíncrona entre la aplicación y los usuarios. Su uso se detalla en el capítulo de desarrollo de la aplicación 4. Una alternativa como herramientas de mensajería es Apache Kafka, dada su alta escalabilidad y uso común en sistemas distribuidos. Sin embargo, se optó por RabbitMQ, debido a que las necesidades de este proyecto no requieren de un volumen de mensajes que justifique una solución basada en Kafka. RabbitMQ ofrece una solución más sencilla de desplegar y administrar, ideal para flujos de trabajo controlados.

Para la gestión de dependencias y el ciclo de compilado del *back-end* se ha utilizado Maven, un gestor de proyectos ampliamente adoptado en el ecosistema Java. Este sistema permite definir el ciclo de vida del proyecto de forma declarativa mediante el archivo `pom.xml`, donde se especifican las dependencias necesarias, sus versiones y otros aspectos clave del proyecto. Su madurez y una curva de aprendizaje razonable facilitaron su adopción, permitiendo automatizar el proceso de compilación y asegurar la correcta gestión de librerías externas.

Sobre esta base de Maven se ha utilizado Spring Boot, un marco de trabajo estándar en Java para la creación de aplicaciones empresariales. Spring Boot proporciona configuraciones automáticas robustas y personalizables junto con un conjunto de dependencias que simplifican la integración de componentes comunes como seguridad y acceso a bases de datos entre otros.

La elección de Spring se fundamenta en su madurez, así como su integración con herramientas como Maven y Docker, y su orientación a aplicaciones con una lógica de negocio claramente definida. Además, dicha plataforma Spring está respaldada por la comunidad, dispone de documentación extensa, y se adapta fácilmente a arquitecturas

modulares.

Gracias a las dos herramientas anteriores se han integrado con fácilidad sistemas de almacenamiento de datos. Se utilizaron dos bases de datos, MongoDB una base de datos no relacional basada en documentos, y TimescaleDB. El uso de estás herramientas es detallado y justificado en el capítulo de Desarrollo de la aplicación 4.

La contraparte del *back-end*, el *front-end*, se construyó con Vue.js, una plataforma de desarrollo progresiva diseñada para crear interfaces reactivas de manera sencilla en JavaScript y Typescript. Vue.js se basa en una estructura de componentes reutilizables, donde cada parte de la interfaz está asociada a datos observables. Estos datos, al cambiar, actualizan la vista sin necesidad de recargar la página.

Se emplearon módulos complementarios de Vue.js compatibles como Vue.js Router, para la navegación entre páginas, Pinia, para el control del estado global de la aplicación, y I18n, para la internacionalización de la interfaz. Los detalles técnicos de su implementación se abordan en el capítulo 4.

Complementando Vue.js, para la representación visual de datos y resultados generados por los modelos predictivos, se utilizo la librería Chart.js, una librería ligera que permite generar visualizaciones en tiempo real a partir de datos actualizados dinámicamente. Su integración directa con Vue.js facilito la representación visual de predicciones y estadísticas sin necesidad de soluciones más complejas y pesadas.

Finalmente, para facilitar la integración de los modelos entrenados mencionados anteriormente, se utilizó Kaizten Task Lite[1], una herramienta proporcionada por la entidad colaboradora. Esta aplicación, también desplegada en contenedor Docker, permitió encapsular los algoritmos como servicios accesibles simplificando su incorporación en la aplicación. Su uso fue clave para conectar la parte analítica del proyecto con la capa visual y operativa destinada a los usuarios finales.

3.3.2. Herramientas para el análisis y modelado de datos

Durante el desarrollo de los modelos predictivos se utilizaron librerías del ecosistema Python orientadas a la manipulación de datos, entrenamiento de modelos y visualización. Estas herramientas fueron empleadas desde las fases de preparación hasta la experimentación final.

Para las tareas iniciales del proceso de transformación de datos, generación de variables y los análisis de datos, se utilizó una combinación de Pandas, Numpy y Matplotlib. Pandas y Numpy permitieron estructurar y manipular de manera eficiente el conjunto de datos complementandose entre ellos, mientras que Matplotlib facilitó la representación visual para comprender los datos, fases intermedias del preprocesamiento y resultados de los modelos entrenados. Algunas de las gráficas generadas estan presentes en capítulos posteriores.

Una vez realizados los pasos iniciales de procesamiento. La libreria utilizada para la construcción de los modelos fue scikit-learn, que permitió definir estructuras conocidas como tuberias de procesamiento (*pipelines*). Estas tuberías de procesamiento conectan los distintos pasos del flujo de datos, desde su transformación hasta la predicción final. Su uso garantiza que los mismos pasos aplicados durante el entrenamiento se mantengan , reduciendo errores y mejorando la mantenibilidad del código.

[1]Kaizten Task Lite, microservicio desarrollado por Kaizten Analytics para desplegar modelos de ML como APIs REST. No disponible públicamente.

Aunque Scikit-learn incluye herramientas básicas para la codificación de variables categóricas, estas resultaron insuficientes para ciertos casos. Para complementar dicha codificación se utilizó la librería Category Encoders para aplicar técnicas alternativas que permitan codificar variables categóricas.

Con el objetivo de almacenar los *pipelines* con sus modelos entrenados para uso posterior sin necesidad de repetir el proceso de entrenamiento, se utilizó cloudpickle, una herramienta de serialización avanzada. A diferencia de módulos estándar como Pickle o Joblib, esta biblioteca permite guardar objetos más complejos, incluyendo funciones personalizadas o clases definidas dinámicamente. Esto resultó especialmente útil al guardar *pipelines* que contenían transformadores definidos manualmente, lo que permitió su reutilización directa dentro de la aplicación.

Una vez entrenados unos modelos iniciales. Se procedió a utilizar Hyperopt para la optimización de hiperparámetros. Esta herramienta permite definir espacios de búsqueda y automatizar la selección de combinaciones de parámetros mediante técnicas de búsqueda bayesiana, reduciendo la necesidad de ajustes manuales tras una configuración inicial.

Finalmente, en cuanto a los modelos utilizados, se exploraron diferentes enfoques y se optó por el uso de los modelos de Scikit-learn y de XGBoost, los cuales son desglosados en el capítulo 6.

3.4. Normativa y legislación

El presente trabajo se ha visto afectado por el marco de protección de datos personales, recogido en la Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales[2]. En su artículo 1 se establece:

> *Adaptar el ordenamiento jurídico español al Reglamento (UE) 2016/679 del Parlamento Europeo y el Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de sus datos personales [...] Garantizar los derechos digitales de la ciudadanía conforme al mandato establecido en el artículo 18.4 de la Constitución.*
>
> —Ley Orgánica 3/2018, de 5 de diciembre.

Durante el desarrollo de este trabajo se utilizaron registros históricos de accesos al recinto portuario, en los cuales se incluyen identificadores personales como el Documento Nacional de Identidad (DNI). En cumplimiento con lo dispuesto por la legislación se adoptaron medidas orientadas a proteger la confidencialidad de la información tratada. Estas son:

- El uso exclusivo de los datos con fines académicos.
- El procesamiento de los datos en entornos controlados y no públicos.
- La codificación y anonimización de los identificadores personales, durante el preprocesamiento, impidiendo la reidentificación de personas físicas.

[2]Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. Publicada en el BOE nº 294, de 6 de diciembre de 2018.

Capítulo 4

Desarrollo de la aplicación

Este capítulo describe en detalle el proceso de construcción de la aplicación software desarrollada como parte de este trabajo. La aplicación, concebida como una solución *full-stack*, tiene como objetivo principal facilitar la gestión operativa de accesos en entornos portuarios, integrando además modelos de predicción de tiempos de estancia y volumen de tráfico vehicular.

El desarrollo se abordó mediante una metodología iterativa-incremental, lo que permitió incorporar progresivamente funcionalidades clave a lo largo del proyecto, adaptándose a nuevas necesidades surgidas durante el análisis y entrenamiento de los modelos predictivos. A través de una arquitectura multicapa, se diseñaron e implementaron los distintos componentes de la aplicación, incluyendo la capa de presentación (interfaz de usuario), lógica de negocio, persistencia de datos y servicios auxiliares como autenticación y mensajería.

Se detallan los requisitos funcionales y no funcionales recogidos, el diseño estructural y técnico de la solución, y finalmente, la implementación de cada una de las capas que componen el sistema. Asimismo, se incluye la integración de herramientas externas para autenticación, mensajería y persistencia, que dotan a la aplicación de capacidades robustas. Este enfoque ha permitido construir una herramienta operativa eficaz, capaz de visualizar información histórica, gestionar accesos portuarios y mostrar resultados derivados de modelos de machine learning de forma accesible para usuarios técnicos y no técnicos.

4.1. Análisis de requisitos del software

Antes de abordar el desarrollo técnico de la aplicación, resulta imprescindible realizar un análisis de los requisitos funcionales y no funcionales, los cuales definen el comportamiento de la aplicación. Este análisis parte de una comprensión del entorno operativo portuario y de las necesidades de los distintos perfiles de actores presentes. En la siguiente sección se describe el modelo de dominio que estructura la lógica de la aplicación, se identifican los casos de uso y actores que interactúan con el sistema, y se enumeran los requisitos no funcionales clave que condicionan su arquitectura y diseño.

La solución propuesta se concibe como una plataforma web orientada a dar soporte a la operativa portuaria relacionada con la gestión de accesos y predicciones de tráfico de camiones. Desde el análisis inicial se identificó la necesidad de una arquitectura en dos capas claramente diferenciadas: una capa de presentación accesible a través de navegador

web (*front-end*) y una capa lógica encargada de gestionar la operativa y los datos (*back-end*).

4.1.1. Clases del modelo del dominio

A nivel de modelado, se identificaron una serie de entidades fundamentales que forman el núcleo de la aplicación. Entre ellas se encuentran el puerto, el camión, el conductor, la reserva y el acceso, cuya estructura y relaciones se resumen en la figura 4.1. Estas entidades representan los elementos básicos de la operativa portuaria. Por ejemplo, el acceso de un vehículo al recinto queda representado por un evento que recoge la puerta por la que accede, el conductor, el vehículo y el momento en que ocurre. Las reservas, por su parte, definen la planificación previa de esa entrada, permitiendo asociar a un camión con un conductor, un horario estimado y una zona del puerto para utilizar mientras se espera al buque. La existencia de esta estructura facilita tanto la trazabilidad como la explotación posterior de los datos.

Estas entidades no actúan de forma aislada, sino que mantienen entre sí una estructura relacional coherente que permite representar con precisión el funcionamiento del sistema portuario. En particular, una reserva se vincula siempre a un camión y, opcionalmente, a un conductor, indicando un intervalo de tiempo estimado para la estancia del vehículo en el recinto. Un acceso, por su parte, representa la entrada o salida efectiva de un camión en el puerto, asociándose también con un conductor, una puerta y una compañía si corresponde. Cada puerto está compuesto por un conjunto de puertas, cada una con su propia geolocalización y tipo, lo que permite distinguir entre accesos de entrada, salida o mixtos. Este modelo relacional ha sido diseñado para soportar tanto la operativa diaria como la trazabilidad histórica, y es la base sobre la que se construyen las funcionalidades de la aplicación.

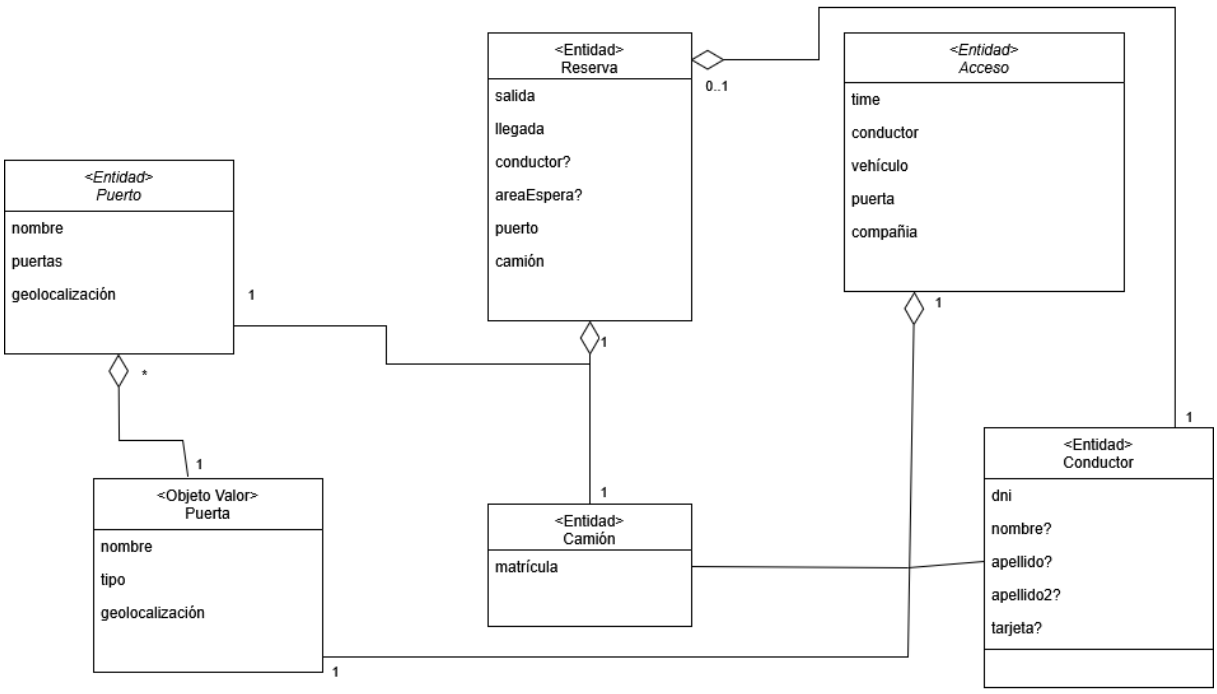

Figura 4.1: Diagrama de clases de relaciones.

A nivel individual, cada entidad presenta una estructura interna específica, detallada en la figura 4.1, que permite representar su papel dentro de la lógica de la aplicación.

La entidad conductor contiene atributos como DNI, número de tarjeta y datos personales opcionales. Un conductor representa a un usuario responsable de conducir camiones. Esta información permite identificar de forma única a cada conductor dentro de la aplicación y vincularlo a accesos o reservas, lo que hace posible asociar cada interacción portuaria con una persona concreta.

La entidad camión representa un vehículo identificado mediante matrícula. Es una entidad fundamental en la aplicación, ya que está directamente vinculada tanto a las reservas planificadas como a los accesos registrados. Su inclusión permite hacer un seguimiento individualizado del recorrido de cada vehículo en el recinto portuario.

La entidad puerto incluye atributos como nombre y localización geográfica, y se relaciona con un conjunto de puertas. Estas puertas se dividen funcionalmente en puntos de entrada o salida. Esta entidad permite modelar la infraestructura física del recinto, y su definición correcta es imprescindible para otras operaciones.

La entidad reserva permite vincular un camión y un puerto con una futura entrada y salida del recinto. Puede asociarse también a un conductor y a una zona de espera. Su papel es facilitar la planificación anticipada de accesos al puerto, permitiendo al gestor portuario anticiparse a situaciones de congestión o conflicto de horarios.

Finalmente, la entidad acceso representa eventos reales de entrada o salida efectuados por los camiones. Se asocia a una puerta concreta, a un camión y a un conductor, y contiene información temporal similar a la reserva. Puede incluir además la empresa vinculada al transporte. Esta entidad es clave para el análisis posterior del comportamiento del tráfico portuario y para validar las reservas realizadas.

Este modelo de dominio proporciona una base estructural sólida para el desarrollo de la aplicación, permitiendo tanto la operación diaria como la explotación analítica posterior de los datos históricos. Además, establece las relaciones y entidades necesarias para soportar funcionalidades de predicción, visualización y control de accesos de forma coherente y escalable.

4.1.2. Actores y casos de uso

Cada una de las entidades presentes en el dominio (conductores, puertos, accesos, reservas, etc.) requiere operaciones básicas de creación, lectura, modificación y eliminación, comúnmente conocidas como operaciones CRUD. Estas operaciones son necesarias para mantener actualizada la aplicación y responder a los cambios operativos del entorno. Por ejemplo, la gestión de accesos contempla tanto la inserción manual de eventos como la consulta de registros históricos. Del mismo modo, la funcionalidad de reservas exige una interfaz que permita generar nuevas reservas, actualizarlas o cancelarlas en función de las necesidades del momento. Estas operaciones deben estar sujetas a control de permisos en función de distintos roles, de forma que se garantice tanto la seguridad como la trazabilidad de las acciones realizadas.

Para ello se identificó como uno de los requisitos funcionales la necesidad de gestionar a los distintos usuarios que harán uso de la plataforma. Esta gestión implica tanto el registro de usuarios como la distinción entre diferentes tipos de roles, cada uno con permisos específicos, que permiten el uso de funcionalidades concretas. Se contemplan, entre ellos, perfiles como el de administrador, gestor portuario y conductor de camiones. Los administradores deben tener acceso completo a todas las funcionalidades de la aplicación,

incluyendo la creación, lectura, modificación y eliminación de puertos, accesos, reservas, conductores y camiones. Los gestores portuarios, por su parte, tienen acceso limitado a la consulta de datos, gestión de accesos y reservas. Los conductores deben poder consultar sus propias reservas y, en función del diseño, solicitar nuevas o modificarlas si están dentro de los márgenes permitidos.

Esta jerarquía de roles permite un control preciso de qué información y acciones están disponibles para cada tipo de usuario. Por ejemplo, mientras que un gestor puede consultar los registros de acceso de todo el puerto y crear nuevas reservas para distintos vehículos, un conductor sólo debe tener acceso a sus propios datos, respetando principios de privacidad y minimización de privilegios.

Adicionalmente, todos los roles tienen acceso a las funcionalidades CRUD asociadas a la entidad camión, al ser un recurso común en la operativa. En particular, los conductores deben poder registrar manualmente un acceso cuando se produce su entrada en el puerto, ya que la aplicación no se encuentra actualmente integrada con sensores físicos o sistemas automáticos de control. Esta funcionalidad permite mantener actualizados los registros de actividad, incluso en ausencia de infraestructura automatizada en el recinto portuario.

Dado que la aplicación trabaja con datos estructurados y con cierta complejidad, se requiere de mecanismos de persistencia adecuados para almacenar tanto los datos operativos como los históricos. La existencia de bases de datos que respalden todas las entidades mencionadas se considera esencial desde las fases iniciales del análisis, no solo para mantener la información actualizada, sino también para poder realizar operaciones estadísticas y visualizaciones a partir de los datos registrados.

Adicionalmente, se identificó la necesidad de ofrecer una interfaz clara y útil para gestores que deban analizar la actividad dentro del recinto portuario. Para ello, la aplicación debe ofrecer una forma accesible y comprensible de visualizar tanto los accesos históricos como las predicciones generadas por los modelos. Estas visualizaciones deben resumir la información clave para la toma de decisiones, como el número de accesos por franja horaria o la distribución por puertas del puerto.

Un requisito adicional identificado durante el análisis es la necesidad de integrar un mecanismo de carga masiva de datos. En el entorno portuario, donde los registros de acceso se pueden generar desde sistemas externos o en grandes volúmenes, es imprescindible que se permita a los administradores importar archivos estructurados con información de accesos. Este sistema de importación debe facilitar la integración de la plataforma con los procedimientos existentes en los puertos, garantizando así una transición fluida hacia el uso de la herramienta.

Asimismo, se ha considerado recomendable, desde el punto de vista funcional, contar con un sistema de notificaciones internas que facilite la interacción entre la aplicación y los usuarios. Estas notificaciones no solo mejoran la experiencia de uso, sino que permiten informar de eventos relevantes como confirmaciones de reserva o cambios de estado en las entidades, los cuales pueden llegar a afectar a gran parte de la operativa.

Con todas estas funcionalidades, la plataforma ofrece a cada tipo de usuario una serie de herramientas específicas que responden a sus necesidades operativas dentro del entorno portuario. Además del análisis textual de los requisitos, se ha considerado útil representar las acciones de cada rol mediante diagramas de casos de uso, que permiten visualizar de forma estructurada las funcionalidades asociadas a cada tipo de usuario dentro de la aplicación.

Estos diagramas se han construido siguiendo la jerarquía de roles definida en la aplicación: todo lo que puede hacer un conductor también puede hacerlo un gestor portuario,

y todo lo que puede hacer un gestor puede hacerlo un administrador. De este modo, se evita la repetición de casos de uso comunes entre roles y se representa únicamente aquello que es específico de cada uno.

En la figura 4.2 se presenta el conjunto de funcionalidades disponibles para el conductor, incluyendo la gestión de sus reservas y camiones, así como la consulta de predicciones para la toma de decisiones. El gestor portuario, representado en la figura 4.3, accede además a funcionalidades de análisis de accesos y consulta de datos operativos. Finalmente, la figura 4.4 muestra los casos de uso exclusivos del administrador, incluyendo la gestión completa de puertos y la importación masiva de datos al sistema.

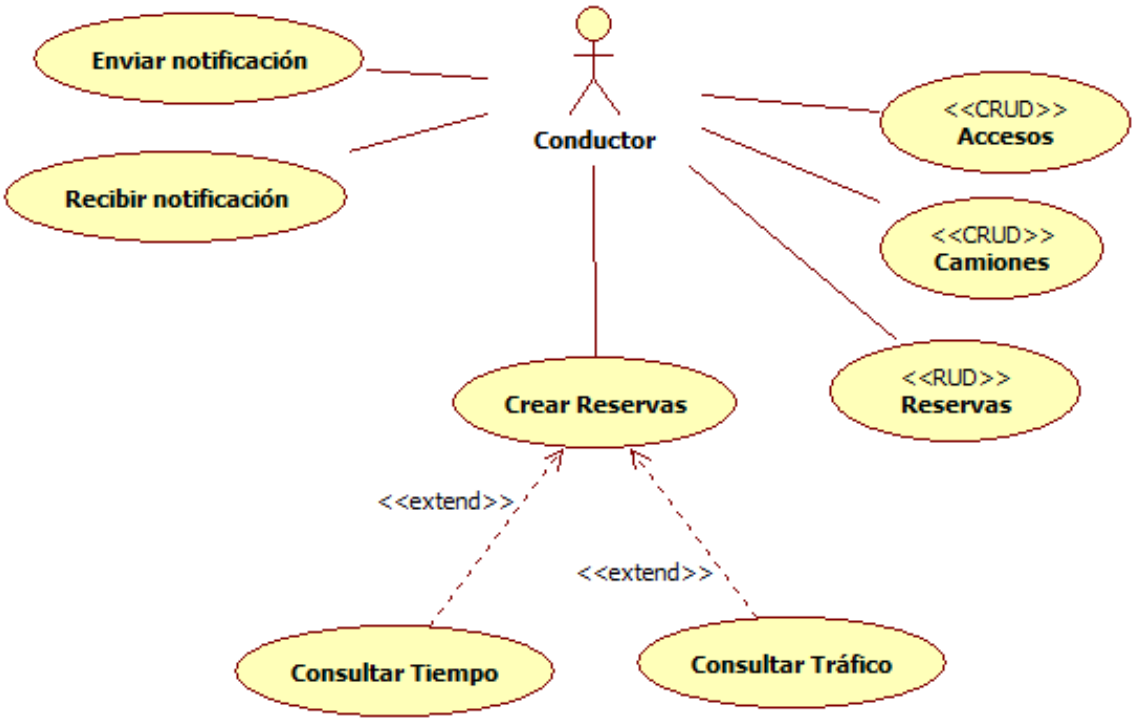

Figura 4.2: Diagrama de casos de uso del conductor.

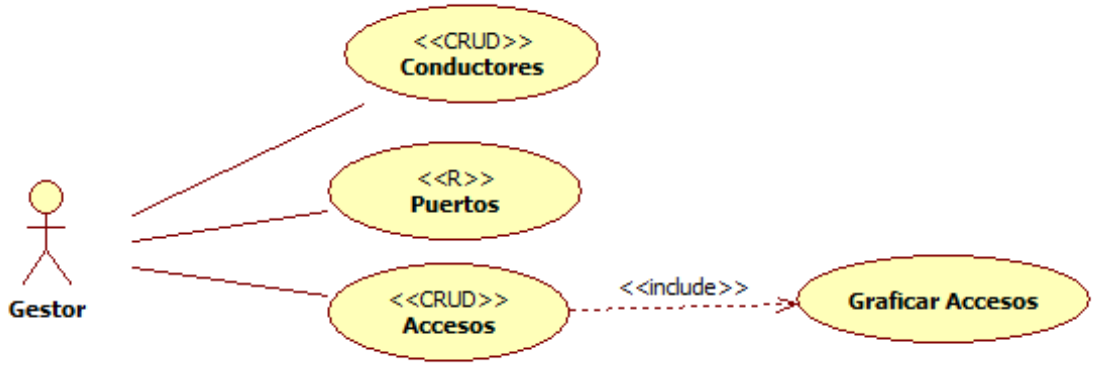

Figura 4.3: Diagrama de casos de uso específicos del gestor.

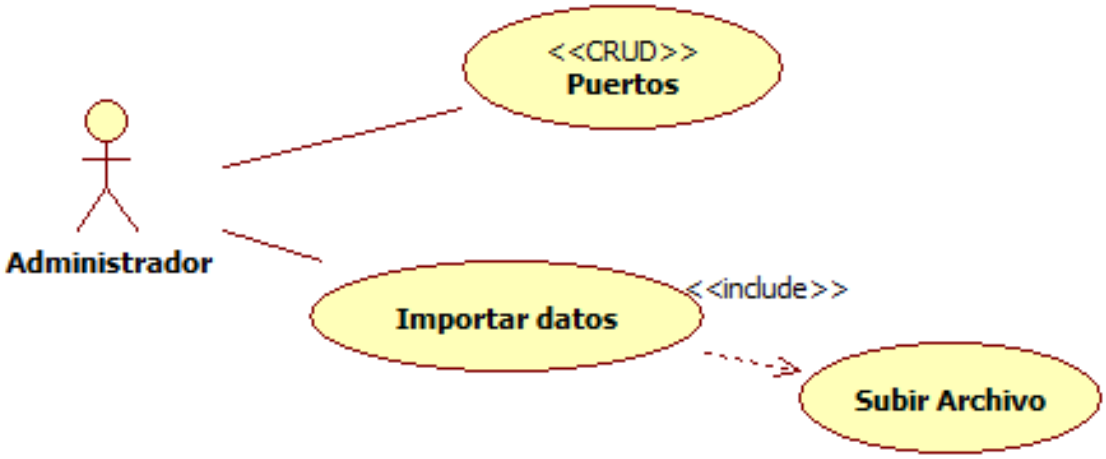

Figura 4.4: Diagrama de casos de uso específicos del administrador.

4.1.3. Requisitos no funcionales

Además de los aspectos funcionales descritos, se identificaron también una serie de requisitos no funcionales que condicionan la arquitectura y el comportamiento general de la aplicación.

Desde la perspectiva de requisitos no funcionales, se estableció como prioritario que la aplicación sea modular, fácilmente mantenible y adaptable a distintos entornos. La posibilidad de desplegarla en distintos puertos con configuraciones específicas y de ampliarla con nuevos módulos sin rediseñar la aplicación en su conjunto se considera un aspecto central. También se ha identificado la necesidad de asegurar la privacidad de los datos tratados, especialmente en lo relativo a los identificadores personales de los conductores, lo que implica una correcta gestión del acceso y del tratamiento de dicha información.

Por otro lado, uno de los pilares fundamentales de la solución propuesta es la capacidad de utilizar modelos de aprendizaje automático (ML) para ofrecer predicciones sobre el tiempo estimado de estancia de los camiones en el recinto portuario. Desde el análisis de requisitos, se identificó la necesidad de que la aplicación cuente con un mecanismo que le permita comunicarse de manera transparente con estos modelos, sin acoplarse a su lógica interna ni requerir conocimiento de su funcionamiento específico.

Para ello, se requiere una forma de integración modular y desacoplada que permita encapsular los modelos entrenados como componentes externos accesibles, por ejemplo, a través de servicios o interfaces bien definidas. Esta integración debe permitir a la aplicación consultar los resultados de predicciones en base a los datos operativos que se busquen definir (reservas), sin importar cómo esté implementado internamente el modelo. De este modo, se garantiza que en el futuro puedan utilizarse otros modelos más precisos o actualizados, sin necesidad de modificar el núcleo de la aplicación.

La finalidad de esta integración es apoyar operativamente a los distintos roles. Por ejemplo, las predicciones generadas pueden servir como herramienta para que los gestores identifiquen picos de congestión o zonas saturadas, y que los conductores puedan consultar estimaciones de tiempo de estancia previas a realizar su entrada, permitiendo así una planificación más eficiente. Funcionalidades como consultar salida o consultar tráfico deben facilitar este acceso, mostrando al conductor información útil basada en los resul-

tados de los modelos. Para cumplir estos objetivos, la lógica de la aplicación debe ser capaz de traducir las respuestas de los modelos en información comprensible y útil para los usuarios, presentándola mediante gráficos, tablas o métricas agregadas.

Esta capacidad predictiva se convierte así en un requisito funcional más, que complementa al resto de funcionalidades básicas descritas anteriormente, y que requiere una planificación estructural desde las fases tempranas del diseño de la aplicación. Su valor reside no solo en la funcionalidad inmediata que ofrece, sino en que permite que la plataforma evolucione con el tiempo en base a los avances en modelos de ML, sin necesidad de rediseñar la aplicación desde cero.

Por último, aunque algunas funcionalidades no han sido desarrolladas en la versión actual de la aplicación, ya se han considerado en esta fase de análisis de requisitos. Entre ellas, se encuentra la integración con sensores físicos o sistemas de control de accesos automáticos, y la implementación especifica para que los conductores puedan acceder a sus accesos. Todas estas funcionalidades podrían formar parte de futuras versiones, pero su previsión desde esta fase permite que la aplicación actual esté preparada estructuralmente para soportarlas más adelante sin grandes modificaciones.

4.2. Diseño del software

Desde el punto de vista funcional, la aplicación se organiza siguiendo una estructura en capas, donde cada una cumple con una responsabilidad específica dentro del sistema. Esta organización permite desacoplar las distintas partes del software, mejorar su mantenibilidad y facilitar su evolución futura.

La **capa de presentación** es la encargada de la interacción con el usuario. Su responsabilidad principal es recoger datos de entrada, transmitirlas a la capa de lógica de negocio y mostrar los resultados generados. En este diseño, la presentación actúa como consumidora de los servicios ofrecidos por las capas internas.

La **capa de lógica de negocio** representa el núcleo funcional del sistema. Aquí se aplican las reglas del dominio portuario, se definen de manera estricta los casos de uso y se gestiona la coordinación de las operaciones sobre las entidades. Esta capa es independiente de la forma en la que los datos son presentados o almacenados.

La **capa de persistencia** es responsable de almacenar y recuperar los datos necesarios para el funcionamiento del sistema. Incluye los mecanismos de acceso a bases de datos, así como la transformación entre objetos del dominio en el negocio y estructuras persistentes.

Finalmente, la **capa de servicios externos** encapsula la integración con otros servicios ajenos a la aplicación principal, como el sistema de autenticación, los modelos de predicción (ML) y el servicio de mensajería o notificaciones. Esta separación permite gestionar estos componentes de forma aislada y sustituirlos o extenderlos sin alterar el resto de la aplicación.

4.2.1. Arquitectura del software

La arquitectura seleccionada para el desarrollo de la aplicación es la **arquitectura hexagonal** [24], también conocida como arquitectura de *puertos y adaptadores*. Este enfoque permite desacoplar la lógica de negocio de los mecanismos externos, como las interfaces de usuario, las bases de datos o los sistemas de mensajería, que pueden cambiar con el tiempo. Esta separación facilita la evolución de la aplicación, permitiendo incorporar nuevas funcionalidades de forma progresiva sin afectar al núcleo del sistema.

A diferencia de otras arquitecturas como Modelo-Vista-Presentador (MVP) [25] o Modelo-Vista-Modelo de Vista (MVVM) [26], que están más orientadas a la organización de la capa de presentación, la arquitectura hexagonal pone el foco en el aislamiento del dominio respecto a sus dependencias externas.

Su estructura se organiza en tres componentes principales: *dominio*, *aplicación* y *adaptadores*. El *dominio* contiene las reglas de negocio fundamentales; la *aplicación* orquesta los casos de uso; y los *adaptadores* actúan como interfaces con el exterior. El diseño promueve una dirección de dependencia desde el exterior hacia el interior (de los adaptadores al dominio), de modo que añadir o sustituir adaptadores sea una tarea sencilla, mientras que modificar el núcleo del dominio requiera una justificación significativa.

Esta disposición se puede visualizar como un hexágono, cuyo orden desde el centro hacia el exterior es: *dominio*, *aplicación* y finalmente *adaptadores*.

El dominio contiene las clases principales de la aplicación. Estas se han dividido en entidades y objetos valor [27]. Una entidad es una clase cuyos objetos son modificables y representan elementos con entidad propia, teniendo varias interfaces que aportan usos relacionados con dicha entidad. Un objeto valor, por el contrario, mantendrá su valor desde que su creación, de manera similar a un objeto primitivo como una cadena de texto inmutable, se puede llamar a un *setter* que devolverá un nuevo objeto con los cambios pertinentes al mismo. Una ventaja de esto es que permite la seguridad de utilizar objetos valor en colecciones con multihilos de manera segura, pues no nos encontraremos que el objeto valor en memoria ha cambiado de manera inesperada.

El componente de aplicación de la arquitectura está formado a partir de casos de uso y repositorios. Cada caso de uso está implementado por un servicio, y cada entidad del dominio tiene repositorios asociados que permiten acceder o modificar sus datos. Esta capa coordina la lógica que la aplicación necesita para ejecutar sus funcionalidades desacoplando los detalles técnicos de infraestructura.

Adaptadores, hay de distintos tipos y todos ellos forman parte de la capa exterior de la aplicación, ya sea debido a que puede haber distintas maneras de resolver un problema particular, como si es por la dependencia a una herramienta externa que se pueda llegar a cambiar en un futuro. Algunas de las funciones de este componente es implementar los controladores http y la persistencia, permitiendo a los servicios utilizar dichos adaptadores para implementar los casos de uso sin que estos tengan que conocer los detalles de implementación.

4.2.2. Diseño de la capa de negocio

La capa de negocio representa el núcleo funcional de la aplicación y está organizada conforme a los principios de la arquitectura hexagonal. Su diseño promueve una separación clara entre el modelo de dominio, los casos de uso y los mecanismos de entrada y salida, con el fin de lograr un sistema modular de arquitectura hexagonal, mantenible y fácilmente extensible.

El componente central de esta capa es el *package domain*, que encapsula los conceptos del dominio portuario. Aquí se definen los elementos fundamentales que componen la lógica del negocio, agrupados en paquetes como *entity*, *valueObject* y *enumerate*. Esta organización permite representar de forma estructurada las entidades clave del sistema, así como los atributos que deben mantenerse inmutables y los tipos que definen las reglas de negocio. El propósito de este paquete es modelar de forma fiel la realidad operativa del entorno portuario, sin introducir dependencias con otras capas o tecnologías.

En un nivel superior se encuentra el *package application*, que contiene los mecanismos para orquestar los distintos casos de uso definidos en los requisitos. Este paquete se subdivide en componentes como *usecase* y *service*, los cuales encapsulan las operaciones funcionales del sistema desde la perspectiva de negocio. También incluye interfaces de repositorio y abstrae servicios como el sistema de notificaciones. La capa de aplicación actúa como intermediaria entre el dominio y el exterior, coordinando las acciones que los usuarios pueden realizar sin acoplarse a detalles técnicos.

Los adaptadores se encuentran organizados en el *package adapter* y representan el punto de contacto entre el sistema y el entorno. Estos se dividen en adaptadores de entrada, que reciben peticiones (como el módulo REST), y adaptadores de salida, que interactúan con recursos externos. Cada adaptador implementa únicamente la lógica necesaria para traducir entre los formatos externos y el modelo interno, asegurando así que la lógica de negocio permanezca aislada.

La comunicación con bases de datos documentales y de series temporales, la carga de archivos estructurados y la integración con servicios o módulos de predicción se gestiona mediante paquetes dedicados dentro de *adapter*, lo que permite su evolución sin modificar las capas internas. Por su parte, el *package configuration* centraliza los parámetros de configuración e incialización de los componentes externos, agrupados por funcionalidad: base de datos, algoritmos, mensajería, seguridad y configuración REST.

Este enfoque modular permite mantener una estructura lógica coherente, donde cada paquete tiene una responsabilidad bien definida. La lógica de negocio queda así completamente desacoplada de los detalles de infraestructura, lo que favorece tanto la mantenibilidad como la capacidad de realizar pruebas unitarias o incorporar nuevas funcionalidades de forma progresiva.

4.2.3. Diseño de la capa de presentación

La capa de presentación de la aplicación se ha estructurado de forma modular mediante un conjunto de carpetas claramente diferenciadas, cada una de ellas con responsabilidades bien definidas dentro del sistema. Esta organización permite mantener un diseño, donde cada parte de la interfaz está alineada con los principios del modelo de dominio y desacoplada de los mecanismos internos de la lógica de negocio.

En primer lugar, la carpeta *domain* encapsula el modelo de datos utilizado en el cliente. Al igual que ocurre en el backend, esta carpeta se divide en tres subcarpetas: *entity*, *valueobject* y *enumerate*. Esta organización permite reutilizar conceptos del dominio en la interfaz, manteniendo consistencia semántica entre cliente y servidor, y facilitando el trabajo colaborativo entre equipos de desarrollo front-end y back-end.

La carpeta *application* organiza los componentes lógicos que estructuran el comportamiento de la interfaz. Contiene, por ejemplo, los repositorios que gestionan el acceso a fuentes externas de datos mediante HTTP, así como los casos de uso y servicios que encapsulan operaciones específicas como la creación de reservas o la importación de accesos. Esta carpeta funciona como capa intermedia entre la lógica de presentación y el modelo, lo que favorece un diseño desacoplado y fácilmente testeable.

La carpeta *adapter* representa el punto de conexión entre la interfaz de usuario y los servicios externos. Dentro de ella se encuentran dos módulos principales: *http* y *vuejs*. El módulo *http* contiene los adaptadores que permiten la comunicación con el servidor. Está organizado por entidad, e incluye tanto los repositorios HTTP como los objetos de transferencia (DTOs) agrupados en carpetas específicas, separando claramente los datos

de entrada y salida. También se define aquí el repositorio de notificaciones y el componente de integración con sistemas externos de mensajería.

Por otro lado, la carpeta *vuejs* contiene los elementos visuales de la aplicación. Aquí se estructuran las *views* principales, los *components* reutilizables, las definiciones de rutas en *router*, y los *stores* que encapsulan el estado de la aplicación usando el sistema de gestión de estado de Pinia. Además, se incluye una carpeta *locales* destinada a los archivos de internacionalización, que permite la adaptación del contenido de la interfaz a diferentes idiomas y contextos regionales.

Esta organización modular, basada en carpetas con responsabilidades claramente delimitadas, garantiza que la capa de presentación pueda evolucionar sin afectar negativamente al resto de la arquitectura. Asimismo, refuerza la coherencia del diseño entre cliente y servidor, al seguir principios similares tanto en la estructuración de los modelos como en la gestión de los casos de uso.

4.2.4. Diseño de la capa de persistencia

El diseño de la persistencia responde a la necesidad de gestionar distintos tipos de información generada en el entorno portuario, diferenciando entre datos estructurados y datos temporales. Desde el análisis inicial se identificó que no todos los datos generados por la aplicación comparten la misma naturaleza ni el mismo patrón de acceso, lo que motivó la adopción de una estrategia de almacenamiento utilizando dos soluciones distintas.

Por un lado, se encuentran las entidades estructuradas del dominio, como conductores, camiones, puertos y reservas. Estas entidades presentan una estructura jerárquica, con múltiples atributos y relaciones internas que deben mantenerse coherentes. Por su carácter general y persistente, estos datos requieren un almacenamiento que ofrezca flexibilidad en su estructura, permitiendo almacenar listas anidadas, atributos opcionales y relaciones entre entidades sin incurrir en un alto coste de modelado. En el diseño del sistema, estas entidades se han asociado a colecciones independientes, alineadas conceptualmente con cada tipo de recurso, lo que permite organizar los datos de forma clara y modular.

Por otro lado, se encuentran los registros de accesos al recinto portuario. A diferencia de las entidades anteriores, estos datos representan eventos que ocurren en momentos concretos del tiempo y se acumulan progresivamente. Además, su principal valor no reside en cada evento individual, sino en la posibilidad de analizarlos de forma agregada: por franjas horarias, por puerta de entrada o por zonas de espera. Estas características hacen que un modelo de almacenamiento orientado a series temporales sea más adecuado para su gestión. Este modelo permite optimizar consultas por rango temporal, realizar análisis estadísticos y conservar el rendimiento incluso cuando el volumen de datos crece de forma significativa.

La elección de esta estrategia de persistencia tiene que ver con una necesidad de eficiencia. El conjunto puede almacenar datos estructurados de forma flexible y consultar grandes volúmenes de eventos temporales sin comprometer el rendimiento. Al mantener separadas ambas responsabilidades, se facilita además la evolución independiente de cada parte.

En conjunto, la capa de persistencia se ha diseñado para adaptarse a las necesidades de la operativa portuaria, garantizando tanto la consistencia y organización de los datos estructurados como el rendimiento y análisis eficiente de los datos temporales. Esta diferenciación permite mantener un diseño coherente que responde adecuadamente a las características del dominio.

Desde el punto de vista estructural, los datos persistidos en la aplicación se organizan en dos grandes modelos de almacenamiento: colecciones de documentos para los datos operativos del dominio, y una tabla de registros temporales para los eventos de acceso.

Las colecciones documentales se corresponden con las entidades principales de la aplicación: conductor, camión, reserva y puerto. Cada colección agrupa documentos con una estructura definida, que incluye tanto atributos primarios como relaciones con otras entidades. Por ejemplo, un documento de tipo *Booking* contiene una referencia al camión implicado, al puerto correspondiente y una franja horaria reservada definida por la llegada y salida. Además, esta estructura admite atributos adicionales, como zonas de espera y conductor asignado, que pueden estar presentes sólo en algunos documentos, gracias a la flexibilidad del modelo documental.

La colección de *Drivers*, por su parte, almacena los datos personales de los conductores, como su nombre, apellidos, DNI, número de tarjeta y empresa vinculada. La de *Trucks* recoge la información básica de los vehículos, como su matrícula. Finalmente, la colección de *Ports* define los distintos recintos portuarios disponibles, incluyendo sus puertas de entrada y salida, que también se almacenan dentro del propio documento del puerto junto a su localización separada en latitud y longitud.

En cuanto a los datos temporales, la aplicación utiliza una tabla dedicada al almacenamiento de accesos reales. Cada fila de esta tabla representa un evento de entrada o salida de un camión por una puerta especifica del puerto. El esquema de esta tabla incluye campos como el identificador del vehículo, el conductor, la puerta utilizada, el sentido del movimiento (entrada o salida), la empresa del conductor y una marca temporal precisa que permite ubicar el evento dentro del historial del sistema. Esta estructura tabular y temporal está diseñada para permitir consultas agregadas (por ejemplo, número de accesos por hora, por zona o por puerta).

Esta diferenciación entre documentos y registros temporales refleja no sólo una decisión técnica, sino también una adecuación al dominio: mientras que las reservas, camiones y conductores constituyen entidades persistentes de la aplicación, los accesos representan eventos efímeros pero fundamentales para el análisis. Al separar ambos modelos de almacenamiento, el diseño permite optimizar el acceso a cada tipo de dato según su naturaleza, respetando los principios de coherencia y rendimiento de la aplicación.

4.3. Implementación

La aplicación desarrollada en esta sección cuenta con un *back-end* desarrollado utilizando Java, mientras que el *front-end* utiliza Typescript. Dado que muchas de las funcionalidades desarrolladas implican tanto al *back-end* como al *front-end* y otros servicios. Se ha optado por una estructura basada en capas para dar a entender la implementación de los componentes de la aplicación.

4.3.1. Implementación de la capa de servicios

Este es un software el cual tiene dividida la funcionalidad principal en distintos contenedores Docker. Los contenedores que no han sido desarrollados durante este trabajo, pero que han sido configurados y son una parte esencial de la aplicación, son: Keycloak, utilizado para la autenticación de usuarios; RabbitMQ, responsable de la mensajería y las notificaciones internas; Kaizten Task Lite, encargado de la ejecución de los algoritmos de aprendizaje automático; y las bases de datos MongoDB y TimescaleDB, donde se

almacena la información estructurada (noSQL) y temporal (SQL), respectivamente.

Por otro lado, los contenedores desarrollados durante este trabajo son el contenedor de *back-end*, que implementa la capa de negocio, y el contenedor de *front-end*, que implementa la capa de presentación. El primero ha sido desarrollado en Java y constituye el núcleo funcional de la aplicación, siguiendo una arquitectura hexagonal que separa las responsabilidades internas de la lógica de negocio y facilita su mantenimiento. Este contenedor interacciona directamente con los servicios de almacenamiento, mensajería y autenticación. El contenedor del *front-end*, por su parte, está desarrollado en TypeScript y permite la interacción visual del usuario con la aplicación, estructurándose también según los principios de modularidad definidos anteriormente.

La figura 4.5 presenta un resumen visual de la arquitectura a nivel de contenedores. El flujo de uso comienza cuando un usuario accede a la capa de presentación. Esta realiza una redirección al contenedor de Keycloak para gestionar el proceso de autenticación. Una vez validadas las credenciales, Keycloak emite un JSON *web token* (JWT) que se adjunta a las futuras solicitudes enviadas a la capa de negocio. Dicho token es verificado por el negocio antes de permitir el acceso a las funcionalidades protegidas.

Keycloak se ha configurado con un *realm* específico que encapsula todas las configuraciones de seguridad asociadas a la aplicación. Dentro de este *realm* se han definido tres roles: *admin*, *manager* y *driver*, que reflejan los perfiles funcionales identificados en el análisis de requisitos. Cada usuario es asignado a un rol concreto, y el *back-end* se encarga de extraer esta información desde el JWT. La política de acceso se basa en este rol, garantizando que cada usuario únicamente accede a las funcionalidades que le corresponden. Para ello, se ha implementado un convertidor que transforma las carácteristicas del token en una autoridad válida dentro de la aplicación, asegurando una correcta integración con el modelo de seguridad subyacente.

Por otra parte, la aplicación se comunica con los servicios de ML a través de peticiones HTTP al contenedor Kaizten Task Lite. Este servicio actúa como un orquestador de tareas asincrónicas relacionadas con predicciones. Cuando se necesita obtener una estimación de tiempo de estancia o congestión, la capa de negocio invoca este servicio mediante una petición POST, enviando los datos operativos relevantes como parámetros dentro de un archivo. Kaizten Task Lite reenvía la tarea a un script conteniendo el modelo previamente entrenado, obtiene el resultado y lo devuelve al negocio. Este resultado es entonces procesado y presentado a la capa de presentación en un formato comprensible para el usuario utilizando gráficas. Esta integración está diseñada para ser desacoplada, de manera que se pueda modificar o sustituir el modelo predictivo sin que la lógica del negocio deba alterarse.

En cuanto a las notificaciones, RabbitMQ actúa como intermediario asíncrono entre las capas de negocio y presentación. Cada vez que se produce un evento relevante sobre una entidad, como la creación o modificación de una entidad, la capa de negocio publica un mensaje en la cola correspondiente. Existen colas separadas para las entidades más importantes: *booking.notifications*, *driver.notifications*, *truck.notifications* y *port.notifications*. En el caso de la entidad *Access*, se ha decidido no generar notificaciones, dado que el volumen de eventos podría ser excesivo y generar una carga innecesaria.

La capa de presentación permanece suscrita a las colas mencionadas, y al recibir los mensajes correspondientes las vistas mostradas al usuario incluyen estas notificaciones. Esto permite una experiencia de usuario en la que el usuario cuenta con una manera de saber sobre posibles cambios que hayan ocurrido en la aplicación.

De forma transversal, la configuración de todos estos componentes se incluye en el

archivo de *docker-compose* los archivos necesarios para parametrizar correctamente todos los servicios. Esto incluye la definición del *realm* en Keycloak, las rutas de los servicios, las credenciales para conectarse a las bases de datos, la configuración de RabbitMQ y las URLs de los servicios de predicción. Esta centralización de parámetros garantiza que los entornos de despliegue puedan configurarse de forma coherente y reproducible.

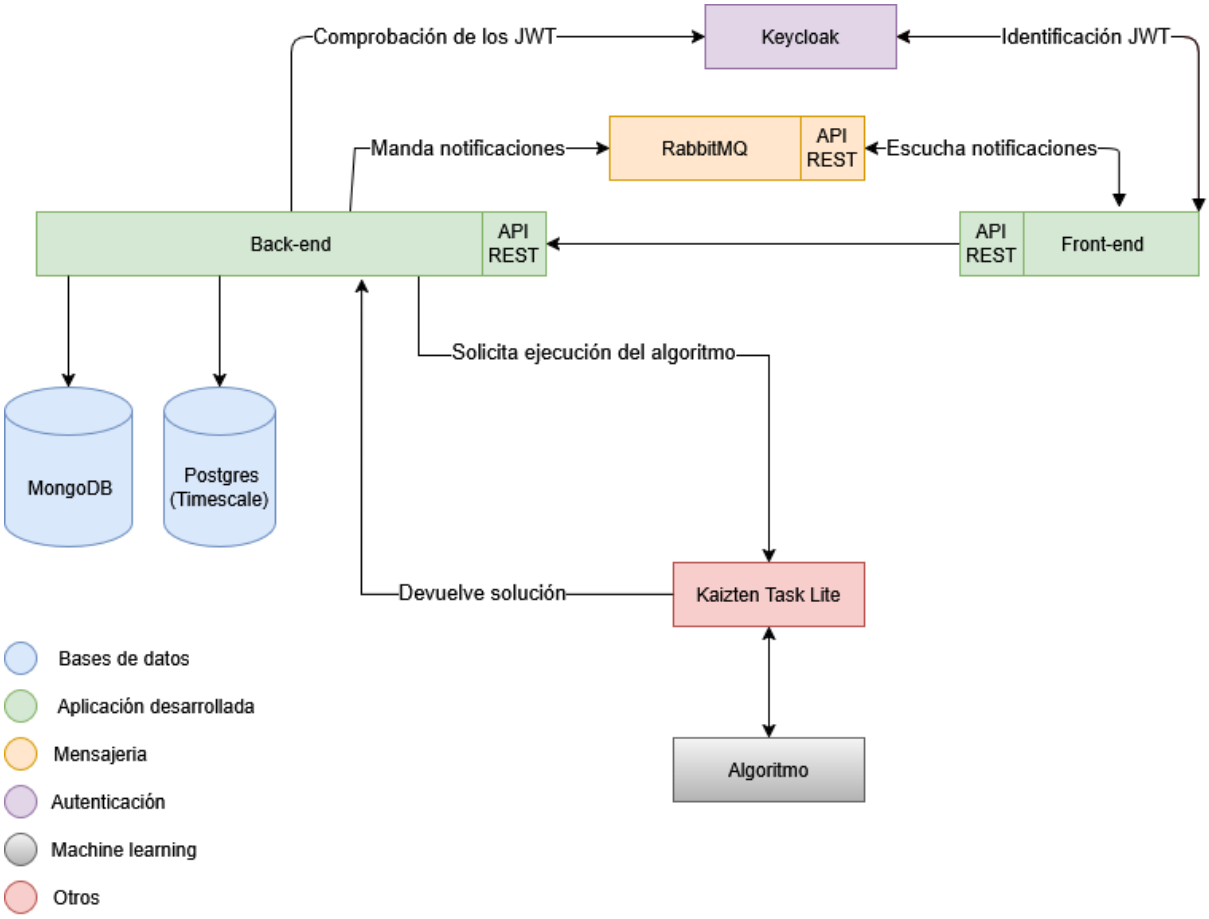

Figura 4.5: Diagrama de componentes de la aplicación.

4.3.2. Implementación de la capa de negocio

La capa de negocio se implementó siguiendo estrictamente la arquitectura hexagonal previamente descrita, asegurando una separación clara entre el núcleo de dominio, los casos de uso y los adaptadores externos. Esta separación se refleja en la estructura modular del código Java, donde cada paquete representa una responsabilidad concreta del sistema.

El núcleo del sistema reside en el dominio, contenido en el paquete *com.kaizten.project .domain.entity*. Aquí se implementan las entidades principales de la aplicación, como *Access*, *Booking*, *Driver*, *Port* y *Truck*. Estas entidades encapsulan los conceptos clave del dominio portuario y contienen su lógica interna, así como los mecanismos de validación necesarios para preservar su consistencia en los objetos valor correspondientes. Junto a ellas se encuentran los objetos valor definidos en *domain.valueobject*, como *CardNumber*, *Plate*, *DniOrNie* o *PortName*, los cuales encapsulan atributos inmutables con significado propio y aseguran que los datos compartidos entre componentes mantengan su integridad. También se incluyen tipos enumerados en *domain.enumerate*, como *Role* y *GateType*, que

refuerzan la claridad del dominio.

Por encima del dominio, la capa de aplicación se implementa en varios paquetes. En *com.kaizten.project.application.usecase* se encuentran las clases que representan los casos de uso del sistema, cada una encargada de coordinar una funcionalidad específica como la creación de reservas, la importación de accesos o la actualización de conductores. Estas clases por lo general no contienen gran lógica técnica, sino que actúan como orquestadores del comportamiento definido en los requisitos.

Los servicios que implementan dichos casos de uso se encuentran en *application.service*. Cada servicio contiene la lógica específica necesaria para llevar a cabo un caso de uso concreto, coordinando la interacción entre las entidades del dominio y los repositorios que gestionan el acceso a datos. Esta separación facilita la reutilización de la lógica en distintos puntos de la aplicación. Las interfaces de repositorio, definidas en *application.repository*, abstraen el acceso a los datos. Existen interfaces independientes para cada una de las entidades principales, como *AccessRepository* o *TruckRepository*, permitiendo a los servicios interactuar con la información persistente sin depender de una tecnología concreta. También en este paquete se encuentra la interfaz *Notificator*, que define una abstracción para el envío de mensajes sin acoplarse a una solución específica de mensajería.

Además de estas funcionalidades centrales, la capa de negocio mantiene comunicación con adaptadores externos que permiten extender el comportamiento de la aplicación sin alterar el núcleo. Uno de los adaptadores clave es el de mensajería, implementado en *adapter.rabbitMQ.rabbitMQNotificator*. Este componente traduce los eventos de la aplicación en mensajes asíncronos enviados a través de RabbitMQ, permitiendo notificar a los usuarios o a otros sistemas de eventos relevantes sin generar dependencias rígidas. La lógica de notificación permanece encapsulada, permitiendo sustituir RabbitMQ por otro sistema en el futuro si fuera necesario.

Otro adaptador relevante es el que se encuentra en *adapter.algorithmn.taskLiteUtilities*, que permite comunicar la aplicación con el motor de predicción externa mediante, para dicha comunicación se utilizan documentos crudos. Este componente transforma los datos operativos del dominio en consultas para los modelos de aprendizaje automático, y procesa las respuestas de dichos modelos para que puedan ser utilizadas por los servicios de la aplicación. Esta separación permite cambiar el modelo de predicción sin impactar la lógica de negocio.

También se implementó un adaptador para la carga de datos masiva mediante archivos CSV. Localizado en *adapter.file.csvFileReader*, este componente transforma los datos contenidos en archivos estructurados en objetos del dominio, permitiendo que la aplicación realice importaciones desde fuentes externas como sistemas portuarios heredados.

Gracias a esta estructura, los adaptadores se comunican con la lógica de negocio mediante interfaces bien definidas, permitiendo una independencia total respecto a las decisiones tecnológicas. Esto garantiza que la aplicación sea extensible y mantenible, preparado para incorporar nuevas funcionalidades sin comprometer su solidez ni su claridad arquitectónica.

4.3.3. Implementación de la capa de presentación

La capa de presentación se ha desarrollado utilizando TypeScript sobre la plataforma de desarrollo Vue.js, organizando el código de forma modular y coherente con los principios de la arquitectura hexagonal. Aunque este entorno no utiliza paquetes al estilo Java, se ha seguido una estructura basada en carpetas que cumple un propósito análogo, dividiendo

claramente las responsabilidades y manteniendo la separación entre el dominio, los casos de uso, los adaptadores y los componentes visuales.

En primer lugar, el directorio *domain* agrupa los conceptos fundamentales del modelo portuario. Al igual que en el *back-end*, se han definido aquí las entidades *Access*, *Booking*, *Driver*, *Port* y *Truck* dentro de la subcarpeta *entity*. Estas clases encapsulan los datos y comportamientos básicos asociados a cada elemento del dominio, como reservas, accesos o vehículos. Esta replicación del dominio en el frontend permite mantener coherencia con la capa de negocio y facilitar el intercambio de información entre cliente y servidor.

Junto a las entidades, se han definido objetos valor en la carpeta *valueobject*, como *CardNumber*, *Plate*, *Gate*, *DniNie* o *PortName*, que encapsulan atributos inmutables y con significado propio dentro del dominio. También se incluye una enumeración en *enumerate/gate-type.ts* para representar los tipos de puerta definidos por la aplicación. Este enfoque facilita el modelado de datos con texto explícito y mejora la seguridad durante el desarrollo, al reducir el uso de cadenas o tipos genéricos.

La carpeta *application* implementa la capa de orquestación, análoga a la aplicación en backend. En ella se encuentran los casos de uso dentro de *usecase*, como *create-booking.ts*, *import-accesses.ts* o *update-driver.ts*, cada uno de los cuales encapsula una operación concreta del sistema, basándose en los requisitos funcionales definidos previamente. Los servicios que materializan estos casos de uso se encuentran en *service*, donde se coordinan las llamadas a los repositorios y la lógica necesaria para cada funcionalidad. Por su parte, las interfaces de acceso a datos se encuentran en la carpeta *repository*, que define contratos para interactuar con las fuentes de datos desde el frontend.

Para representar resultados paginados, se ha creado la clase *Page*, que permite gestionar de forma uniforme los listados largos de accesos, conductores o camiones, tanto en la vista como en las operaciones de recuperación de datos.

Los adaptadores de presentación se organizan dentro de la carpeta *adapter*. En particular, el subdirectorio *http* contiene los adaptadores que permiten interactuar con la API REST expuesta por el backend. Cada entidad tiene su propio repositorio HTTP, como *access-http-repository.ts* o *truck-http-repository.ts*, que encapsula las llamadas al servidor. Además, se han definido objetos de transferencia de datos (DTO) en la subcarpeta *dto*, organizados en ficheros como *access-post-json-request.ts* o *port-json-response.ts*, los cuales estructuran de forma clara los datos que se envían y reciben mediante las interfaces HTTP. Este enfoque permite una separación clara entre los datos que maneja la aplicación y la estructura de las peticiones externas.

En la carpeta *vuejs* se concentran los elementos visuales y de interacción con el usuario. Aquí se encuentra la carpeta *components*, que agrupa los componentes Vue reutilizables, mientras que las principales pantallas de la aplicación se implementan en *views*. La navegación entre estas vistas está gestionada en *router*, y la lógica de estado global se almacena en *stores/store*, utilizando Pinia como sistema de gestión de estado. Además, para soportar múltiples idiomas, se incluye una carpeta *locales* que contiene los archivos necesarios para la internacionalización de la interfaz de usuario.

Esta organización modular facilita la comprensión de la aplicación y el mantenimiento de los componentes. Cada carpeta cumple una función bien definida dentro de la arquitectura, permitiendo introducir nuevas funcionalidades a la aplicación en el futuro sin introducir complejidad innecesaria ni romper la cohesión del diseño inicial.

4.3.4. Implementación de la capa de persistencia

La capa de persistencia de la aplicación se ha diseñado con el objetivo de ofrecer una solución eficiente para el almacenamiento y recuperación de datos, aprovechando las características específicas de cada tipo de información gestionada por la aplicación. Para ello, se ha optado por una solución que integra dos tipos de bases de datos diferenciadas: una base de datos documental (MongoDB) y una base de datos de series temporales (TimescaleDB). Esta decisión permite optimizar el tratamiento de datos estructurados y eventos cronológicos, adaptando la aplicación a las necesidades del entorno portuario.

MongoDB se utiliza como base de datos principal para almacenar información estructurada relacionada con las entidades del dominio, tales como *Access*, *Booking*, *Driver*, *Port* y *Truck*. Cada una de estas entidades se representa como una colección independiente dentro de la base de datos. Estas colecciones permiten realizar operaciones CRUD básicas y almacenar documentos con una estructura flexible. Para cada colección se han definido tres componentes: un lector, un escritor y una implementación de repositorio. El lector se encarga de extraer información desde la base de datos, el escritor realiza inserciones y actualizaciones, y el repositorio coordina ambas operaciones como interfaz intermedia con la capa de aplicación.

Por su parte, TimescaleDB se encarga de almacenar los accesos reales registrados por los conductores al recinto portuario. Esta base de datos, optimizada para la gestión de datos temporales, permite organizar los accesos como eventos en el tiempo, con atributos como fecha, hora, puerta de entrada o salida, conductor asociado y camión implicado. Su uso permite realizar consultas temporales eficientes y genera la base de datos necesaria para la elaboración de informes estadísticos o visualizaciones. Para su gestión, se ha implementado un adaptador específico compuesto por un repositorio, un DAO (Data Access Object) y un inicializador que define la estructura necesaria para almacenar correctamente los eventos.

La elección de esta doble estrategia de persistencia responde a las características propias del sistema. Las entidades del dominio requieren flexibilidad para evolucionar en su estructura, lo cual se adapta bien a MongoDB. Sin embargo, los accesos al puerto presentan un carácter temporal secuencial que justifica el uso de TimescaleDB. Esta separación permite mantener una base de datos documental ligera y centrada en la operativa, mientras que se delega la carga de análisis temporal a una tecnología más especializada.

En conjunto, esta capa se integra de manera desacoplada con el resto del sistema. La lógica de negocio accede a los datos exclusivamente a través de interfaces de repositorio, sin depender de detalles técnicos de persistencia. Esto asegura que futuros cambios en el almacenamiento, como migraciones a otros motores de base de datos o reestructuraciones de colecciones, puedan realizarse sin necesidad de modificar la lógica central del sistema.

Capítulo 5

Conjunto de datos

En este capítulo se describe de forma detallada el proceso completo llevado a cabo para el desarrollo de los modelos predictivos utilizados en este trabajo. Siguiendo la primera parte de la metodología CRISP-DM como marco de referencia, se abordan las fases del ciclo de vida del modelo: desde la adquisición inicial de los datos, pasando por su análisis exploratorio, hasta la preparación, transformación y codificación necesarias para su uso en algoritmos de ML.

A lo largo del capítulo se analizan con detalle las características del conjunto de datos, identificando patrones relevantes, errores estructurales y distribuciones significativas. Este análisis previo resulta esencial no solo para comprender el problema, sino también para ajustar adecuadamente las estrategias de modelado. Para facilitar esta tarea, se incluyen múltiples representaciones gráficas que permiten interpretar los datos como el comportamiento de los modelos entrenados.

5.1. Conjunto de datos de entrenamiento

Tras el comienzo del trabajo, se recibieron una serie de archivos Excel proporcionados por la entidad colaboradora. Estos archivos contienen registros de la actividad del Puerto de Santander desde 2019 hasta 2023. Entre la información contenida se encuentran las escalas operativas de buques, las escalas de mercancías y los registros de control de entradas y salidas al recinto portuario.

El Puerto de Santander, como se describió en la introducción 1 tiene varias puertas de acceso al entorno portuario como la mostrada en la figura 1.1, los vehículos pasan a través de una puerta y registran su acceso al puerto. Estos accesos son los datos de control de interés para los modelos a entrenar.

Los datos seleccionados como base para este proyecto corresponden al control de accesos, puesto que son los datos que presentan una mayor relación con el objetivo del trabajo: estimar la estancia de los camiones en el recinto portuario. El archivo control de accesos está formado por un único Excel. Este archivo Excel contiene una hoja de cálculo por puerta, y cada puerta únicamente tiene una dirección de tránsito (entrada o salida del recinto). La tabla 5.1 enumera los nombres originales de las hojas de cálculo del control de accesos y su relación con las zonas y puertas del puerto encontrados en los datos.

Cada registro incluye información como la marca temporal del acceso, el identificador de la puerta, y campos administrativos como el DNI del conductor, el número de tarjeta y la empresa asociada. Uno de los archivos adicionales, titulado Camiones, contenía estos mismos campos de forma redundante y fue descartado para evitar duplicidades en el

Tabla 5.1: Correspondencia entre hojas de Excel, puertas y áreas del puerto.

Nombre de hoja	**Nombre de la puerta**	**Área**
Entrada Maliaño 1	3 C. MALIAÑO ENTRADA VEHICULOS VIAL 1	Maliaño
Entrada Maliaño 2	65 C. MALIAÑO ENTRADA MALIAÑO VIAL 2 (I/P)	Maliaño
Salida Maliaño	4 C. MALIAÑO SALIDA VEHICULOS	Maliaño
Entrada Raos 1	273 CONTROL RAOS ENTRADA 1	Raos
Entrada Raos 2	265 CONTROL RAOS ENTRADA 2	Raos
Entrada Raos 3	257 CONTROL RAOS ENTRADA 3	Raos
Salida Raos 1	281 CONTROL RAOS SALIDA 1	Raos
Salida Raos 2	289 CONTROL RAOS SALIDA 2	Raos
Salida Raos 3	297 CONTROL RAOS SALIDA 3	Raos
Entrada Antonio Lopez Norte	89 SANIDAD NORTE ENTRADA	Antonio López
Salida Antonio Lopez Norte	97 SANIDAD NORTE SALIDA	Antonio López
Camiones	Nulo	Nulo

procesamiento.

Como se aprecia en las tablas 5.2 y 5.3, el conjunto de datos presenta una estructura heterogénea. Existen inconsistencias en la forma de representar nombres de puertas (mayúsculas, abreviaciones), así como variaciones en el campo de fecha y hora. Aunque este problema no impide el análisis ya que cada hoja utiliza un único nombre coherente sí requiere un proceso de normalización para automatizar su tratamiento. Por ello, se mantuvieron los nombres originales, respetando la nomenclatura utilizada por el Puerto de Santander.

El campo evento, presente en todos los registros, describe el tipo de acción registrada. En la mayoría de los casos indica claramente una entrada o una salida de vehículos, pero también se han detectado otros eventos no operativos como la conexión de un controlador o un error de lectura. Estos registros, que responden a notificaciones técnicas del sistema de acceso, no aportan información útil para el análisis del flujo de vehículos y se consideraron ruido, por lo que fueron filtrados en etapas posteriores del procesamiento.

También se detectaron valores nulos o incompletos en ciertos campos, lo que hace necesaria una fase de limpieza para garantizar la calidad del conjunto. Estos problemas, comunes en datos operativos, dificultan una explotación directa sin transformación previa.

Un punto importante a señalar es la falta de una columna explícita que indique el área del puerto a la que pertenece cada puerta. Aunque esta información puede deducirse por el nombre, no está integrada directamente en los registros. Por ello, se optó por crear una nueva columna denominada área, que agrupa las puertas según su localización (Raos, Maliaño, Antonio López).

Por último, el volumen elevado de registros, junto con su distribución desigual en el tiempo y entre puertas, presenta retos adicionales. Hay una clara concentración de eventos en ciertas zonas o franjas temporales, lo que puede introducir sesgos dificiles de remediar.

- **Fecha**: Momento exacto en el que se registró el acceso.
- **Acceso**: Puerta específica del puerto por la que se realizó la entrada o salida.

Tabla 5.2: Muestra de registros originales de la puerta de entrada Raos 1.

Fecha	**Acceso**	**Evento**	**DNI**	**Nº Tarj**	**Empresa**
26/04/2021 07:20:52	273 CONTROL RAOS ENTRADA 1	Entrance	72...E	900631	APS
26/04/2021 08:22:06	273 CONTROL RAOS ENTRADA 1	Entrance	13...K	13536	CONTRUC. GOMEZ LLOREDA
26/04/2021 08:31:43	273 CONTROL RAOS ENTRADA 1	Entrance	72...Q	18019	TRANSP. RAOS
26/04/2021 09:26:21	273 CONTROL RAOS ENTRADA 1	Entrance	72...D	16587	TTES. MANUEL ANGEL MANRIQUE REAL
26/04/2021 09:51:45	273 CONTROL RAOS ENTRADA 1	Access modified			
29/04/2021 10:59:15	273 CONTROL RAOS ENTRADA 1	CONTROLADORA L512 CONECTADA			
29/04/2021 11:19:29	273 CONTROL RAOS ENTRADA 1	CONTROLADORA L512 DESCONECTADA			
29/04/2021 17:10:37	273 CONTROL RAOS ENTRADA 1	Entrance	72...W	16006	COBASA
29/04/2021 17:25:51	273 CONTROL RAOS ENTRADA 1	Entrance	13...J	15822	MACROPESC
29/04/2021 17:43:32	273 CONTROL RAOS ENTRADA 1	Entrance	72...V	14189	AUTONOMO

- **Evento**: Tipo de evento registrado. En la mayoría de los casos corresponde a una entrada o salida, aunque pueden aparecer registros técnicos como conexiones de controladoras.

- **DNI**: Documento de identidad del conductor que realiza el acceso.

- **Nº Tarj**: Identificador asociado al conductor, utilizado por el sistema de control de accesos del puerto.

- **Empresa**: Entidad a la que está vinculado el acceso.

En resumen, aunque el conjunto de datos presenta una base rica para el análisis, su estructura inicial requiere una preparación cuidadosa. Esta preparación incluye la identificación de inconsistencias, el tratamiento de valores ausentes y la incorporación de información adicional que facilite el análisis posterior. Solo mediante esta limpieza inicial es posible asegurar la calidad y coherencia del *dataset* para fases más avanzadas del desarrollo de las técnicas de ML.

Tabla 5.3: Muestra de registros originales de la puerta de entrada Raos 2.

Fecha	**Acceso**	**Evento**	**DNI**	**Nº Tarj**	**Empresa**
15/04/2021 12:02:01	265 CONTROL RAOS ENTRADA 2	Entrance	72...C	18153	JOAQUIN CARRAL TTES. Y LOGISTICA S. L.
15/04/2021 12:03:49	265 CONTROL RAOS ENTRADA 2	Entrance	13...P	3377	URBASER
19/08/2021 5:02:43		Entrada	71...R	17795	SANTANDER COATED SOLUTIONS
30/10/2021 16:55:18	265 CONTROL RAOS ENTRADA 2	Entrada	76...C	14608	ERHARDT
30/10/2021 16:59:09	265 CONTROL RAOS ENTRADA 2	Entrada	Y2...C	18278	MONTAÑESA DE DESINFECCION
30/10/2021 17:08:15	265 CONTROL RAOS ENTRADA 2	Entrada	72...H	15626	GLOBAL STEEL WIRE
19/08/2021 5:29:24	Entrada	72...Q	4222	APS	

5.2. Limpieza de datos

Tras analizar la estructura original de los archivos proporcionados por el Puerto de Santander, se observó que los datos estaban distribuidos en múltiples hojas dentro de un único archivo Excel, cada una correspondiente a una puerta concreta y a una única dirección de tránsito (entrada o salida). Para facilitar el tratamiento y análisis posterior, se procedió a una reestructuración del conjunto de datos.

Inicialmente, las hojas de cálculo fueron clasificadas según su nombre en dos grupos: hojas de entrada y hojas de salida. A partir de esta clasificación, se consolidaron dos archivos separados: uno que reúne todos los eventos de entrada y otro con los eventos de salida. Este enfoque permitió mantener una separación clara entre ambos tipos de eventos, sin necesidad de inferencias posteriores sobre la dirección del tránsito, y facilitó la aplicación de procesos de limpieza homogéneos por categoría.

A cada uno de los registros se le añadió además una nueva columna de area, extraída del nombre de la hoja de origen, que indica la zona del puerto asociada a la puerta. Esta información resultó útil para el análisis exploratorio de los accesos en fases posteriores.

Durante la exploración inicial del campo evento, se detectaron múltiples valores distintos para representar la misma acción, así como la presencia de registros puramente técnicos que no aportaban valor analítico. Se conservaron únicamente los eventos válidos identificados como *entry* y *exit*, que corresponden a movimientos reales de entrada o salida de vehículos. Otros valores inconsistentes o mal escritos, como *entrada manual*, *entradas*, *entrance*, o cadenas aparentemente aleatorias como *78qlq7rl1* pertenecientes a otros campos, fueron normalizados mediante reglas específicas y asignados correctamente a su columna funcional correspondiente utilizando patrones.

Por otro lado, se eliminaron los registros cuyo evento reflejaba tareas del sistema

o comunicaciones internas, tales como acceso conectado, reader connected, inicio configuración, entre otros. Estos eventos técnicos, frecuentes en sistemas automatizados, no representan acciones humanas ni movimientos de vehículos y, por tanto, se consideraron ruido. Asimismo, otros eventos como Acceso no autorizado o Falta número de tarjeta, que no eran homogéneos ni presentaban una semántica clara para su interpretación operativa, fueron eliminados o reasignados a categorías válidas si existía información suficiente para hacerlo con certeza.

Se prestó atención adicional a la validez de los valores en los campos: fecha, DNI, tarjeta y empresa. Para ello, se implementaron expresiones regulares que permitieran identificar patrones válidos de DNI y NIE, y se aplicaron funciones de limpieza sobre registros que presentaban inconsistencias en el orden de las columnas, intercambiando valores entre campos cuando se detectaba desalineación.

En cuanto al campo de fecha, cualquier registro cuya marca temporal no pudiera ser interpretada correctamente fue descartado. Esto permitió asegurar que todos los registros conservados fueran cronológicamente válidos y utilizables en análisis temporales o secuenciales.

Los campos DNI y Empresa fueron completados, cuando fue necesario, con valores por defecto (G75278085 y AUTONOMO, respectivamente), para asegurar que no existieran valores nulos en atributos clave del conjunto. También se validó el campo Acceso, y en aquellos registros donde el valor resultaba nulo o incoherente, se sustituyó por el acceso más frecuente dentro de la hoja correspondiente, permitiendo así preservar la consistencia estructural sin eliminar el registro completo.

Aunque el campo DNI presenta cierta heterogeneidad en su formato y no siempre es completamente fiable, se optó por conservarlo tras su limpieza y normalización, ya que constituye la única referencia disponible en los datos que puede asociar un evento con un conductor de forma consistente. Por este motivo, se trató como un campo clave en el proceso de estructuración, garantizando al menos su completitud mediante valores por defecto cuando fue necesario.

Como parte del proceso de depuración, todos los registros descartados fueron almacenados en un archivo independiente con el fin de mantener un control trazable sobre las decisiones de limpieza aplicadas. Este conjunto incluye eventos descartados, accesos atípicos o registros imposibles de corregir automáticamente.

En total, se generaron dos archivos estructurados que contienen el resultado del proceso: Control entrada.csv, con los eventos de entrada limpios y estructurados, y Control salida.csv, con los eventos de salida equivalentes.

Este procedimiento de limpieza y estructuración permitió obtener un conjunto de datos limpio, homogéneo y listo para ser utilizado en los siguientes pasos del análisis exploratorio y codificación de variables.

Como se observa en la tabla 5.4, el conjunto de datos resultante presenta una estructura homogénea. Todos los registros incluyen una marca temporal válida, una puerta de acceso definida y un área correctamente asignada, lo que evidencia la efectividad del proceso de limpieza y estructuración realizado. Asimismo, se aprecia la normalización del campo evento, en este caso con el valor entry de forma unificada al tratarse del archivo correspondiente a entradas, y la conservación de información administrativa clave como el DNI, el número de tarjeta y la empresa vinculada.

Por otro lado, se detectan algunas repeticiones sucesivas de registros con idénticos valores de DNI y punto de acceso, ocurridos con escasos segundos de diferencia. Este patrón sugiere la presencia de posibles duplicados generados por errores en el sistema

Tabla 5.4: Muestra de registros una vez limpiados de las puertas de entrada

Fecha	**Acceso**	**Evento**	**DNI**	**NºTarjeta**	**Empresa**	**Area**
2019-01-02 07:11:55	3 C. MALIAÑO ENTRADA VIAL 1	Entry	136...	13660.0	PROSETE CNISA	maliaño
2019-01-02 07:35:19	3 C. MALIAÑO ENTRADA VIAL 1	Entry	137...	900173.0	APS	maliaño
2019-01-02 13:50:12	3 C. MALIAÑO ENTRADA VIAL 1	Entry	201...	17855.0	POLICIA	maliaño
2019-01-02 13:52:47	3 C. MALIAÑO ENTRADA VIAL 1	Entry	202...	16824.0	AMARRA DORES	maliaño
02/01/2019 07:49	65 C. MALIAÑO ENTRADA VIAL 2 (I/P)	Entry	721...	17353	ADECCO	maliaño
02/01/2019 07:50	65 C. MALIAÑO ENTRADA VIAL 2 (I/P)	Entry	137...	3194	CANTABRI ASIL	maliaño
02/01/2019 07:50	65 C. MALIAÑO ENTRADA VIAL 2 (I/P)	Entry	137...	900058	APS	maliaño
02/01/2019 07:50	65 C. MALIAÑO ENTRADA VIAL 2 (I/P)	Entry	137...	900058	APS	maliaño
02/01/2019 07:50	3 C. MALIAÑO ENTRADA VIAL 1	Entry	137...	17116	SALVA MENTO	maliaño
02/01/2019 07:50	3 C. MALIAÑO ENTRADA VIAL 1	Entry	137...	17116	SALVA MENTO	maliaño
02/01/2019 07:51	65 C. MALIAÑO ENTRADA VIAL 2 (I/P)	Entry	720...	3429	RUSA	maliaño
02/01/2019 07:51	65 C. MALIAÑO ENTRADA VIAL 2 (I/P)	Entry	137...	4046	FRIG.	maliaño

de lectura del puerto, algo relativamente habitual en entornos con lectores automáticos. Estas repeticiones podrían ser objeto de análisis posterior para establecer criterios de consolidación de eventos o detectar anomalías operativas.

Para finalizar este proceso y poder utilizar los datos con el fin de entrenar los modelos predictivos y poder analizarlos de manera exhaustiva se juntan los conjuntos de entradas y salidas en un único archivo. Este proceso se basa en concatenar los datos de cada conjunto.

5.3. Análisis exploratorio

Como punto de partida del análisis exploratorio, resulta fundamental conocer el volumen de datos disponibles tras el proceso de limpieza. La figura 5.1 muestra la distribución de eventos válidos en el conjunto final, diferenciando entre accesos de entrada y salida.

En total, tras la depuración, se conservaron aproximadamente 1.202.453 registros, de los cuales 718.650 corresponden a eventos de entrada y 483.803 a eventos de salida. Esta cifra se obtuvo a partir de un conjunto original ligeramente mayor, del cual se eliminaron únicamente 3.382 registros. Estas pérdidas, que representan una fracción muy reducida del total (menos del 0,3 %), se debieron principalmente a registros técnicos, incompletos o mal formateados que no aportaban valor analítico ni permitían un tratamiento uniforme. Esta depuración mínima resulta aceptable y necesaria para garantizar la consistencia del conjunto de datos.

La gráfica también permite apreciar que existe un volumen relativamente equilibrado entre entradas y salidas, aunque con una diferencia clara a favor de los accesos de entrada. Esta asimetría no necesariamente implica un fallo en el registro, sino que puede reflejar el funcionamiento logístico habitual del puerto: es común que muchos camiones accedan al recinto y posteriormente abandonen el entorno portuario por vía marítima, lo que explicaría que no aparezcan eventos de salida equivalentes. Esta característica del recinto portuario de los datos debe tenerse en cuenta, ya que sugiere que las puertas de entrada podrían estar sometidas a una mayor presión de tráfico que las de salida, con posibles implicaciones para la planificación y optimización de accesos. Adicionalmente esto implica que si se emparejan entradas con salidas, se perderían una gran cantidad de entradas al puerto pues cada entrada sólo puede tener una salida.

Este gráfico permite constatar que los datos están ahora adecuadamente estructurados para su análisis. La relación entre eventos de entrada y salida será clave en etapas posteriores del procesamiento, especialmente en la estimación de la duración de la estancia de los vehículos en el recinto portuario.

La gráfica de la figura 5.2 permite profundizar en el comportamiento de los accesos según la localización geográfica dentro del puerto, mostrando las cantidades de eventos de entrada y salida desglosadas por área: Antonio López Norte, Maliaño y Raos.

A primera vista, se observa que las zonas de Raos y Maliaño concentran la gran mayoría de los accesos, tanto de entrada como de salida, lo cual es coherente con su relevancia operativa dentro del recinto portuario. En particular, Raos destaca como el área con mayor volumen de actividad, con más de 400.000 accesos de entrada registrados, seguida por Maliaño. Esta distribución concuerda con lo descrito en la introducción (1), ya que ambas zonas acogen operaciones logísticas de mayor escala, como el transporte de mercancías a granel. Por el contrario, la zona de Antonio López Norte presenta una actividad notablemente menor, con menos de 25.000 eventos de entrada, lo que sugiere un uso más restringido a funciones como el tránsito local, servicios internos o accesos de personal.

También se mantiene la tendencia observada en el análisis global: en todas las áreas, la cantidad de entradas supera claramente a la de salidas. Esta diferencia refuerza la hipótesis planteada anteriormente, según la cual una parte significativa de los vehículos —en particular, camiones— accede al recinto portuario pero no realiza una salida por vía terrestre. Este comportamiento es coherente con la operativa de embarque de vehículos rodantes en ferris o buques ro-ro, una práctica común en este tipo de entornos logísticos. Este aspecto tiene implicaciones prácticas: las zonas de entrada, especialmente en Raos,

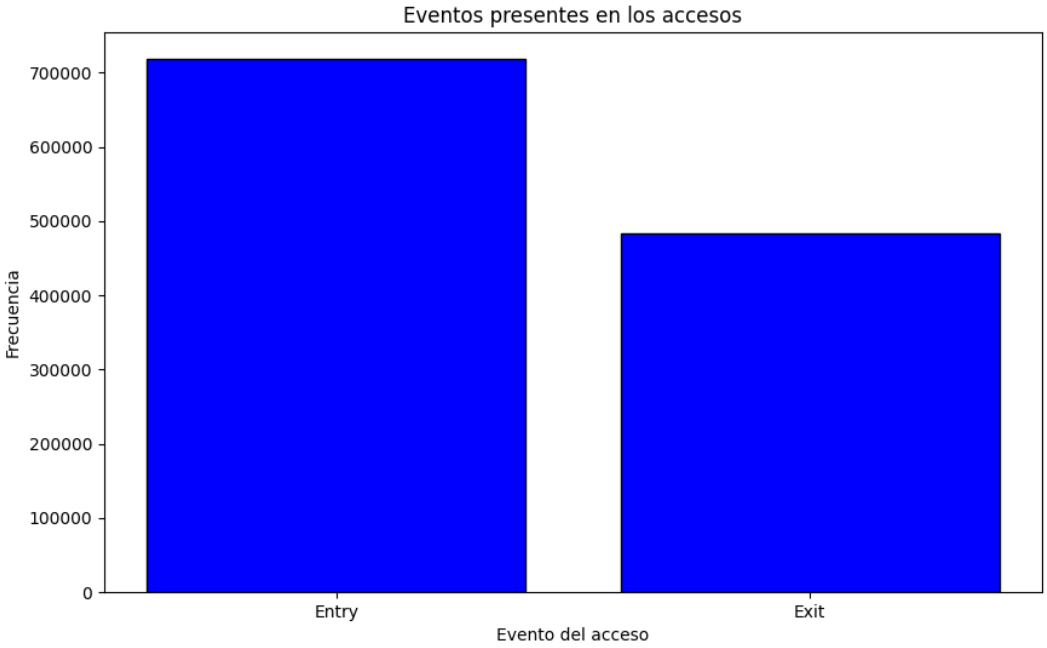

Figura 5.1: Eventos de los accesos de los vehículos registrados en los datos

podrían estar sometidas a una mayor presión de tráfico y requerir una planificación más cuidadosa en términos de infraestructura, vigilancia y fluidez operativa.

Por último, el contraste entre áreas sugiere que la segmentación espacial podría ser un factor relevante a considerar en etapas posteriores del análisis. No obstante, debido al reducido número de puertas por área (entre una y tres en la mayoría de los casos), no se dispone de una muestra suficientemente diversa como para extraer conclusiones estadísticas sólidas por zona. Esta limitación hace que, en esta fase, el tratamiento por puerta individual siga siendo más informativo que una agregación puramente espacial.

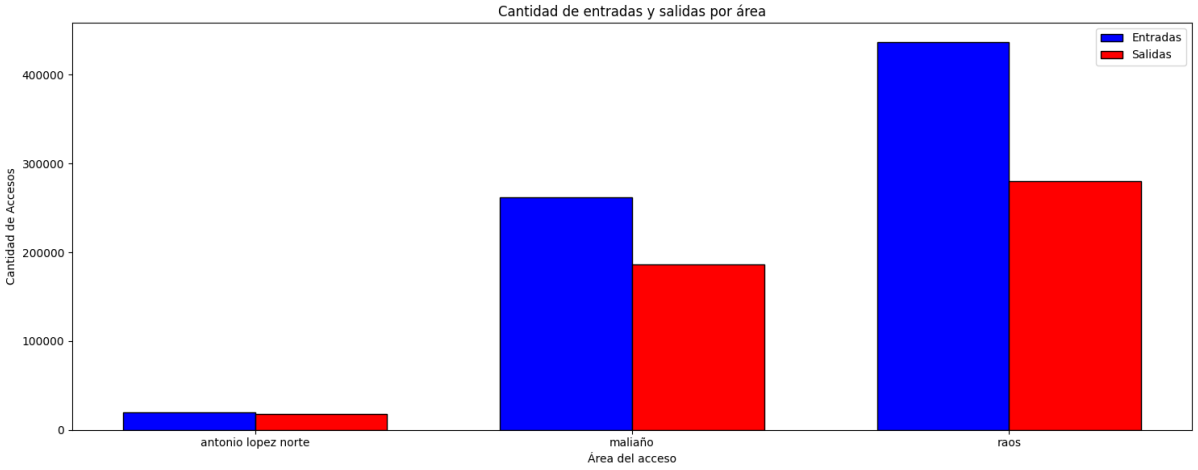

Figura 5.2: Número de registros para cada evento de acceso por área en los datos

A continuación, se presenta una visión temporal de los accesos, comenzando por su distribución anual. La figura 5.3 muestra la evolución del número de registros anuales

en el conjunto de datos. Como se observa, el año 2021 concentra la mayor actividad del periodo, superando los 190.000 accesos, seguido por 2022, mientras que los años 2019 y 2020 reflejan una actividad considerablemente menor.

La caída pronunciada en 2020 puede atribuirse razonablemente a las restricciones logísticas impuestas por la pandemia de COVID-19, que afectó de forma directa tanto al transporte terrestre como al tráfico marítimo. Este comportamiento refuerza la importancia de considerar la variable temporal *año* como posible factor explicativo en los modelos predictivos, ya que los patrones operativos pueden alterarse significativamente por factores económicos, sanitarios o contextuales.

Por otro lado, el repunte de actividad en 2021 podría interpretarse como una fase de recuperación tras la crisis sanitaria, aunque no se puede asegurar con los datos disponibles que dicha tendencia se mantenga de forma estable. De hecho, los registros de 2022 muestran una ligera disminución respecto al año anterior, lo que sugiere cierta inestabilidad estructural en la evolución interanual. Por tanto, aunque el año puede aportar información útil al modelado, no debe emplearse como un predictor aislado, y resulta más fiable cuando se combina con otras variables temporales como el mes o el día de la semana.

En contrapartida, es importante tener en cuenta que el número de años disponibles en el conjunto de datos es limitado, lo que impide obtener una métrica robusta del comportamiento estandar de los accesos portuarios a lo largo del tiempo. Si se pretende aplicar los modelos a años futuros como 2025, el uso explícito del año como variable podría introducir sesgos importantes o pérdidas de precisión, al no haberse entrenado sobre datos representativos de ese nuevo periodo. Este sesgo se atenúa en la medida en que se disponga de un mayor número de años en el conjunto de entrenamiento, ya que una mayor cobertura temporal mejora la capacidad del modelo para generalizar frente a fluctuaciones anuales atípicas como las que se cree que han sucedido en base a la pandemia. En este contexto, podría ser preferible prescindir del año como variable directa o tratarlo con técnicas específicas de generalización temporal.

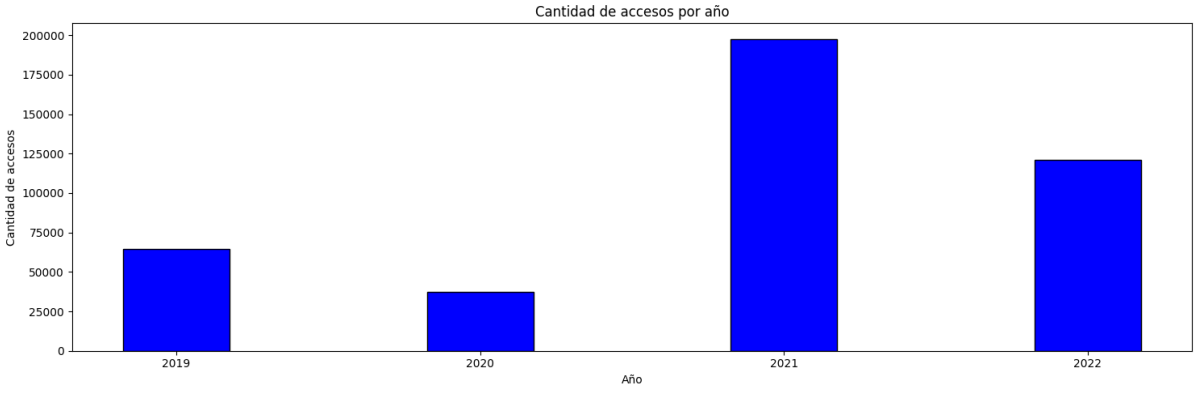

Figura 5.3: Número de registros por año en los datos

La figura 5.4 muestra la distribución mensual de accesos al puerto para cada uno de los años disponibles, permitiendo analizar tanto las variaciones estacionales como las diferencias interanuales. Como ya se observó en el análisis anterior, el año 2021 concentra la mayor actividad general, pero esta gráfica permite apreciar que dicha actividad se concentró especialmente entre los meses de mayo y septiembre, con picos notables en junio y julio.

El año 2020, por el contrario, muestra un comportamiento muy diferente: una caída

pronunciada en los meses de abril y mayo, seguida de una recuperación parcial en verano. Este patrón es coherente con las restricciones de movilidad y actividad económica impuestas por la pandemia durante ese periodo, lo que afectó directamente a la operativa del puerto.

En cuanto a los años 2019 y 2022, se observa una distribución más uniforme de los accesos a lo largo del año, sin picos tan marcados. No obstante, en 2022 sí se identifica una actividad elevada en los primeros meses, que después se estabiliza sin alcanzar los niveles de 2021, lo cual podría indicar un retorno a condiciones más regulares de operación.

Estos resultados refuerzan la idea de que el mes del año puede ser una variable útil en los modelos predictivos, al reflejar tanto ciclos estacionales como efectos externos que impactan en la logística portuaria. Sin embargo, también es importante considerar que estas variaciones mensuales pueden verse influenciadas por factores no comúnes, por lo que su uso en modelos debe acompañarse de otros mecanismos que eviten el sobreajuste a condiciones excepcionales.

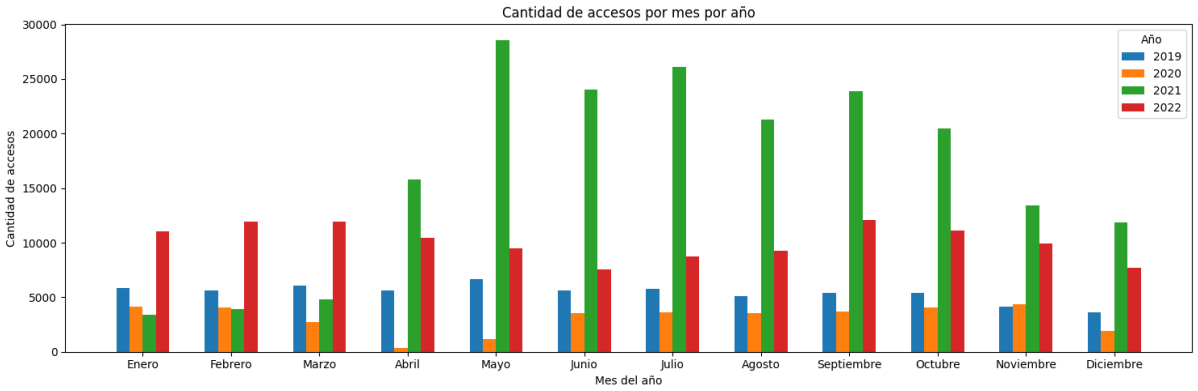

Figura 5.4: Número de registros por mes y por año en los datos

La figura 5.5 representa el número de accesos en función del día de la semana, diferenciando entre eventos de entrada y salida. Se observa que, de lunes a viernes, el tráfico en el puerto es relativamente estable, con volúmenes elevados y consistentes de actividad. No se aprecian grandes fluctuaciones entre días laborables, lo que refleja una operativa regular y continua a lo largo de la semana laboral.

En contraste, los fines de semana presentan una caída en el número de accesos. Este descenso sugiere que durante estos días se reduce considerablemente la actividad logística en el recinto portuario, probablemente por el cierre parcial de instalaciones o la disminución del transporte comercial. Este patrón refuerza la necesidad de tener en cuenta el día de la semana como una variable clave en los modelos predictivos, ya que las condiciones de acceso cambian drásticamente entre días laborales y no laborales.

Un aspecto adicional de interés es que durante los fines de semana, la diferencia entre el número de entradas y salidas se reduce bastante respecto a los días laborables. Esto podría indicar que muchos de los vehículos que acceden al puerto en sábado o domingo realizan operaciones de corta duración y salen ese mismo día, a diferencia de lo que ocurre entre semana, donde es más habitual que esto no suceda.

En conjunto, el análisis semanal revela un patrón operativo claro con implicaciones directas tanto para el modelado predictivo como para la planificación logística en el entorno portuario.

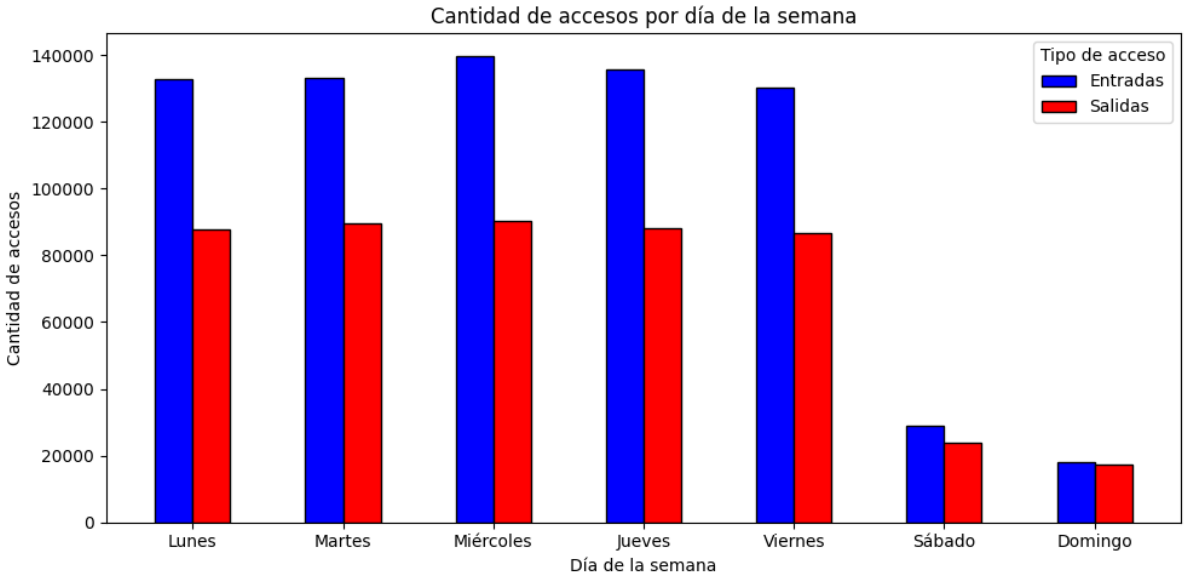

Figura 5.5: Histograma con el número de registros por día de la semana para cada evento en los datos

Por último, la figura 5.6 representa la distribución de los accesos en franjas horarias de una hora. Se observa que la entrada de vehículos comienza a incrementarse a partir de las 5:00, con un ascenso muy pronunciado entre las 6:00 y las 8:00. El máximo se alcanza alrededor de las 8:00, lo que sugiere un inicio temprano de la actividad operativa portuaria. Posteriormente, la cantidad de entradas desciende de forma progresiva a lo largo del día.

Las salidas, en cambio, presentan una distribución más regular y sostenida. Comienzan a incrementarse de forma significativa a partir de las 9:00 y mantienen una presencia constante hasta bien entrada la tarde, alcanzando un punto álgido entre las 13:00 y las 15:00. A partir de ese momento, la actividad disminuye progresivamente hasta las últimas horas del día.

Esta información resulta especialmente útil para identificar franjas de mayor carga operativa, y por tanto potenciales momentos de saturación en las infraestructuras de control. En particular, las primeras horas del día, especialmente entre las 7:00 y las 9:00, concentran el mayor volumen de entradas, lo que puede tener implicaciones logísticas para la planificación de accesos o refuerzo de personal en esas franjas horarias.

En conclusión, el momento del día, en particular, la hora del acceso se revela como una de las características temporales más relevantes del conjunto, tanto por su capacidad explicativa como por su potencial para mejorar la precisión de los modelos predictivos.

Finalmente, cabe mencionar que, debido al elevado número de valores distintos en los campos DNI, numero de tarjeta y empresa, no se realizó un análisis detallado de estas variables de forma individual. No obstante, se considera razonable suponer que pueden influir en la duración de la estancia, ya sea por el tipo de operador logístico, la frecuencia de acceso o la categoría del conductor. Estos factores se tendrán en cuenta en las fases posteriores del modelado.

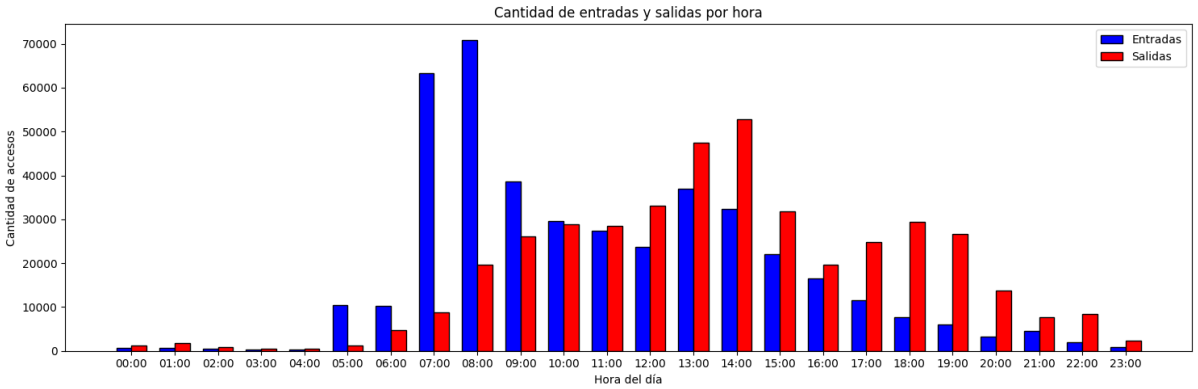

Figura 5.6: Distribución diaria por hora de registros en los datos para cada evento

5.4. Preprocesamiento y emparejamiento de accesos

Una vez completada la fase de limpieza y exploración, se procede al preprocesamiento del conjunto de datos para su uso en modelos de aprendizaje automático. Esta etapa incluye tanto la generación de variables objetivo como la transformación de las variables categóricas y temporales mediante técnicas de codificación. El objetivo es adaptar los datos a un formato numérico y estructurado que permita una correcta interpretación por parte de los modelos predictivos.

Todos los algoritmos de aprendizaje automático requieren que sus variables de entrada estén codificadas numéricamente. Para ello, se aplicaron diferentes estrategias de codificación según el tipo de variable y su número de categorías.

En primer lugar, se consideró el uso de Label Encoding, una técnica básica que asigna un número entero a cada categoría. Sin embargo, este método puede inducir relaciones ordinales artificiales. Por ejemplo, si se codifican las puertas del puerto como (Raos 1 = 0, Maliaño 2 = 1, Maliaño 3 = 6), un modelo podría interpretar erróneamente que existe una distancia o jerarquía entre ellas, cuando no es el caso.

Para evitar este problema, se utilizó principalmente One-Hot Encoding (OHE), que transforma cada categoría en una columna binaria independiente. Este método es especialmente útil cuando el número de categorías distintas es reducido, ya que evita introducir ordenaciones ficticias y permite al modelo tratar las categorías como independientes. No obstante, OHE se vuelve ineficiente o incluso inviable cuando hay un número elevado de categorías, ya que genera una matriz dispersa muy amplia.

En estos casos, como en los campos empresa y DNI, se optó por aplicar Count Encoding, una codificación que reemplaza cada categoría por su frecuencia de aparición en el conjunto de datos. Esto permite capturar indirectamente la familiaridad de una entidad con el puerto, lo cual podría estar relacionado con la duración de las estancias o el comportamiento de acceso.

Por su parte, las variables temporales como la hora, minuto o día de la semana fueron transformadas mediante codificación cíclica[1], empleando funciones seno y coseno para reflejar su naturaleza periódica. De esta forma, se garantiza que, por ejemplo, las 23:00 y las 01:00 estén representadas con valores cercanos, evitando interpretaciones erróneas por parte del modelo. Esta técnica fue preferida frente a la codificación polinómica, que también se exploró inicialmente, pero se descartó por ser menos eficaz en tareas donde se

[1]Una implementación de la codificación cíclica puede consultarse en: `https://www.kaggle.com/code/avanwyk/encoding-cyclical-features-for-deep-learning`

requería una representación continua del tiempo. El propósito inicial de ambas codificaciones era facilitar el uso de modelos lineales y evitar recurrir exclusivamente a árboles de decisión. Sin embargo, incluso en modelos basados en árboles, la codificación cíclica demostró ser ventajosa.

Una vez completado el proceso de codificación, el conjunto de datos fue dividido aleatoriamente en dos subconjuntos: un conjunto de entrenamiento (90 %) y otro de prueba (10 %). Esta división permite evaluar objetivamente el rendimiento de los modelos y evitar problemas de sobreajuste. Aunque la proporción habitual es 80/20, se optó por esta configuración dada la gran cantidad de registros disponibles, lo que garantiza un tamaño de muestra adecuado en ambas particiones.

5.4.1. Tiempo de estancia

Para cada evento de entrada, se buscó la salida más próxima en el tiempo que compartiera el mismo DNI, siempre que ocurriera al menos un minuto después y dentro de un intervalo máximo de 24 horas. Este margen fue definido para excluir emparejamientos inválidos, como aquellos en los que la salida precede a la entrada o donde el desfase temporal es excesivo. El objetivo es que los emparejamientos reflejen únicamente estancias reales dentro del recinto portuario.

En la práctica, las entradas sin salida asociada dentro del intervalo se interpretan como vehículos que abandonaron el puerto por vía marítima, sin pasar por una puerta de salida. De forma análoga, las salidas sin entrada previa pueden deberse a vehículos que accedieron al recinto desde el mar, por ejemplo desembarcando desde un buque.

Este emparejamiento permite calcular la variable objetivo *tiempo de estancia*, expresada en minutos. Dicha variable representa la duración estimada que un vehículo permanece en el recinto portuario y se utiliza posteriormente en el entrenamiento de modelos supervisados.

Como consecuencia directa del proceso, se reduce el número total de accesos utilizables, ya que únicamente se conservan aquellos registros que pudieron ser emparejados correctamente. Tras aplicar esta restricción, se obtuvieron aproximadamente 422.000 pares válidos de entrada y salida, lo que representa una disminución considerable respecto al volumen original. Como es lógico, el conjunto resultante contiene exactamente el mismo número de entradas que de salidas, ya que cada par representa un trayecto completo.

Además del emparejamiento, se analizó la distribución de los tiempos de estancia obtenidos, como se muestra en la figura 5.7. El diagrama revela una variabilidad considerable en los datos, con numerosos valores atípicos que representan estancias inusualmente largas. Esta dispersión supone un reto para los modelos predictivos, ya que dificulta la obtención de estimaciones precisas y consistentes.

A pesar de la alta varianza, se pueden observar ciertos patrones temporales. Las medianas de estancia tienden a seguir una secuencia de picos y descensos a lo largo del día, lo que sugiere que la hora de entrada tiene una influencia sobre el tiempo que un vehículo permanece en el recinto. Este comportamiento se rompe en las franjas nocturnas (especialmente entre las 21:00 y las 05:00), donde los tiempos de estancia se incrementan de forma notable. Esto podría reflejar una operativa logística diferente durante la noche, como esperas prolongadas para embarques o periodos sin actividad operativa.

Este tipo de análisis es clave para justificar el uso de la hora del acceso como variable relevante en los modelos predictivos, aunque también pone de manifiesto la necesidad de técnicas robustas debido a la gran dispersión de datos, los cuales son tan dispersos que

eliminar aquellos que no siguen el patrón común mediante cuantiles eliminaria la mayor parte del conjunto.

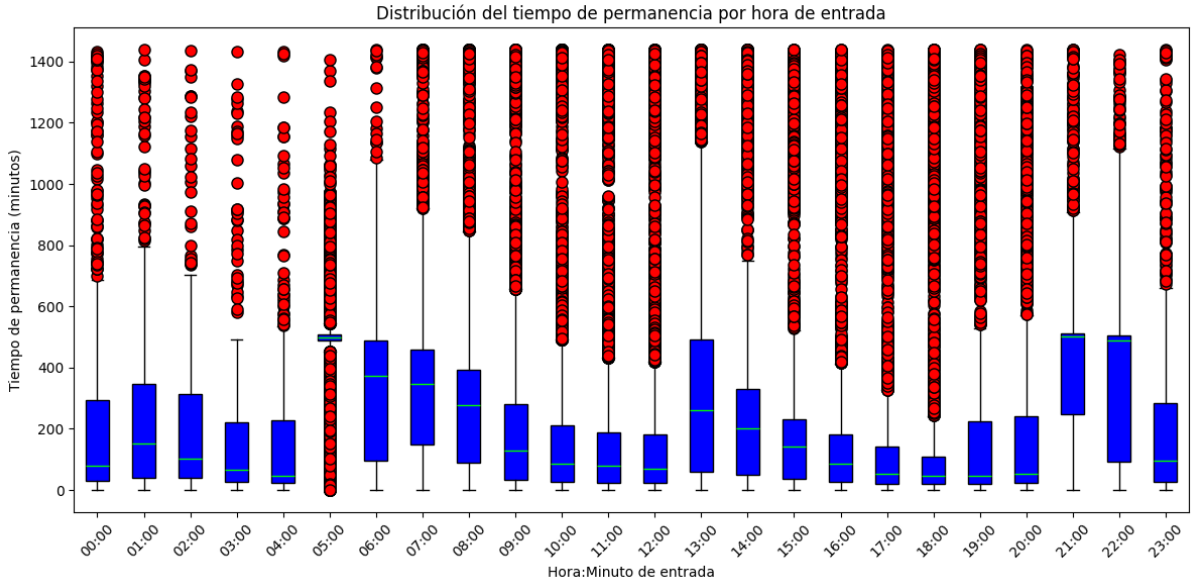

Figura 5.7: Distribución del tiempo de permanencia en minutos de los registros emparejados

Además del tiempo de estancia, se analizó el comportamiento espacial del tráfico entre áreas del puerto a través de los registros emparejados. El objetivo de este análisis es identificar posibles patrones en el flujo de vehículos dentro del recinto, que puedan ser útiles para generar modelos predictivos.

En particular, se estudió la correlación entre las zonas de entrada y salida de los accesos emparejados. La figura 5.8 muestra la cantidad de vehículos que accedieron por cada área y cómo se distribuyeron posteriormente sus salidas en función de la zona. Como se aprecia, existe una fuerte tendencia a que los conductores salgan por la misma área por la que accedieron, lo que sugiere trayectorias locales y lógicas dentro del recinto portuario.

Sin embargo, se identifican excepciones relevantes, especialmente en el caso de la zona de Maliaño. Esta área, aunque concentra una parte significativa de los accesos de entrada, dispone de una única puerta de salida operativa. Esto genera una carga operativa considerable en dicho punto y obliga a una fracción de los vehículos a abandonar el recinto por otras zonas como Raos, lo que podría implicar trayectorias más largas o desviaciones logísticas.

Estos hallazgos confirman que la zona de acceso es una variable con potencial explicativo tanto para el tiempo de estancia como para la estimación del tráfico. Además, ponen de manifiesto la importancia de que en recintos portuarios, no se tiene un control absoluto del tráfico portuario.

Una vez emparejados los accesos y calculadas las variables objetivo, se procedió a la extracción y transformación de las variables explicativas. Dado que los modelos de aprendizaje automático requieren variables numéricas como entrada, las variables categóricas y temporales fueron codificadas siguiendo diferentes estrategias según su naturaleza. Además, se incorporaron nuevas variables derivadas del propio contenido de los atributos, como una clasificación binaria de los conductores según el tipo de documento.

Las principales variables procesadas fueron las siguientes:

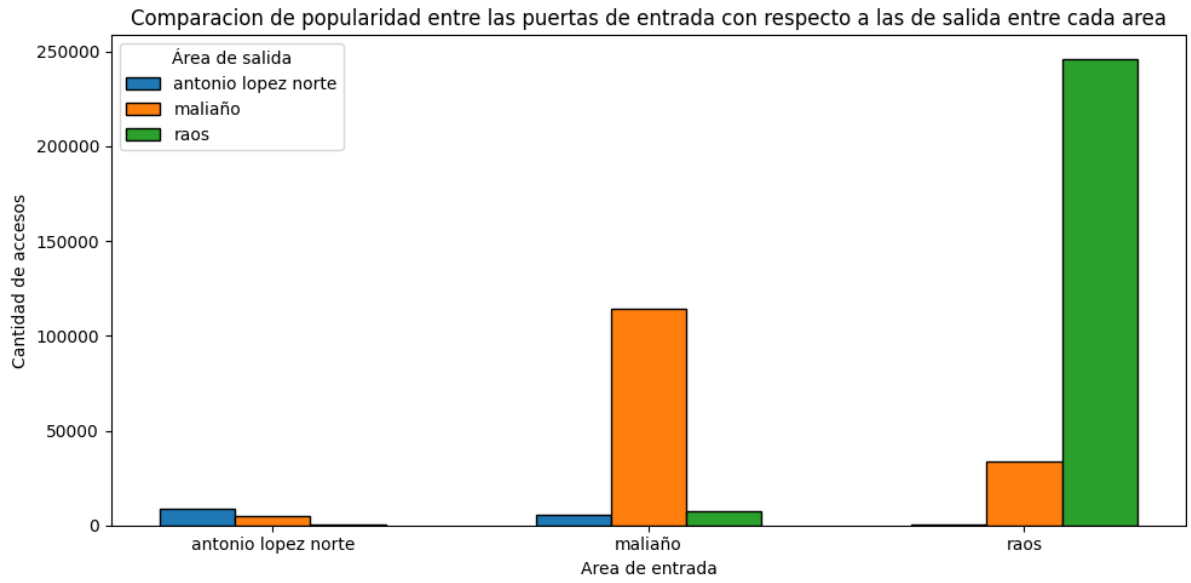

Figura 5.8: Número de accesos emparejados según el área de entrada y la correspondiente área de salida

- **Minutos del Día**: Minutos transcurridos desde medianoche (0–1440). Codificado de forma cíclica mediante funciones seno y coseno.
- **Acceso de entrada** y **Acceso de salida**: Nombres de las puertas utilizadas. Codificados mediante One-Hot Encoding (OHE) por tratarse de categorías discretas sin orden lógico.
- **Día de la semana**, **Mes del año** y **Día del mes**: Variables temporales extraídas de la fecha, transformadas mediante OHE para preservar su carácter categórico sin asumir ordenación artificial.
- **DNI**: Codificado con Count Encoding (frecuencia de aparición), complementado con una variable binaria que indica si el conductor es extranjero, identificando aquellos cuyo documento comienza por X, Y o Z.
- **Empresa**: Codificada mediante Count Encoding para representar la familiaridad operativa de la entidad con el entorno portuario a través de su frecuencia de aparición en los datos.

5.4.2. Tráfico

Además del cálculo del tiempo de estancia, se planteó una segunda línea de análisis centrada en la estimación del tráfico portuario. En este caso, el objetivo no es emparejar accesos, sino predecir cuántos vehículos pasarán por una puerta concreta en una franja horaria determinada. Esta aproximación permite explotar directamente los registros individuales de entrada y salida, sin requerir emparejamiento, lo que evita perder información valiosa en los casos donde no se dispone de ambas marcas.

Para construir esta variable objetivo, se fusionaron los registros de entrada y salida previamente limpiados, y se agruparon por combinación única de fecha, hora y puerta de acceso. A cada grupo se le asignó el número total de accesos registrados, formando así un

conjunto de observaciones donde cada fila representa una franja horaria concreta en una puerta específica del recinto. Se estableció un intervalo de agregación de una hora (por ejemplo, de 09:00 a 10:00), lo que permite capturar la dinámica operativa del puerto con suficiente granularidad.

Con el fin de evitar distorsiones en el modelo, se descartaron del entrenamiento los registros correspondientes al año 2020. Como se ha mencionado en secciones anteriores, este periodo se vio afectado por la pandemia de COVID-19, alterando significativamente los patrones normales de actividad portuaria. Su inclusión habría introducido sesgos atípicos y potencialmente perjudiciales para la generalización del modelo.

La figura 5.9 muestra la distribución del tráfico por hora del día tras aplicar este procesamiento. Se observa una concentración clara de actividad en las primeras horas del día, especialmente entre las 7:00 y las 14:00, lo que concuerda con los patrones diarios identificados anteriormente. Este comportamiento refuerza la utilidad de la hora como variable explicativa relevante.

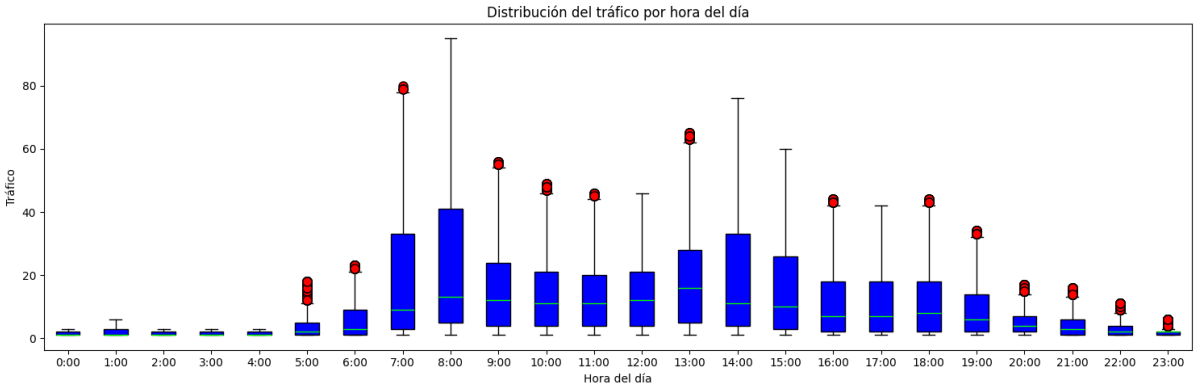

Figura 5.9: Distribución del tráfico presente cada hora de los registros procesados

Para la generación de variables codificadas, se extrajeron múltiples características temporales a partir de la fecha: hora del día, día del mes, día de la semana y mes del año. Todas ellas fueron codificadas mediante funciones seno y coseno para preservar su naturaleza cíclica y evitar errores de interpretación por parte del modelo. Además, se incorporó la puerta de acceso como variable categórica, codificada mediante One-Hot Encoding.

Las variables utilizadas en el modelo de tráfico fueron:

- **Fecha**: Descompuesta en variables temporales (hora, día del mes, día de la semana, mes), todas ellas codificadas de forma cíclica.
- **Acceso**: Identificador textual de la puerta utilizada, codificado mediante One-Hot Encoding.

La variable objetivo es el número de accesos registrados para cada combinación puerta–intervalo horario. Este enfoque permite construir un modelo predictivo más robusto frente a los problemas de emparejamiento, con potencial aplicación en la estimación de carga operativa, planificación de turnos o detección de picos de saturación en el entorno portuario.

Capítulo 6

Modelos de predicción y resultados

En primer lugar, se presentan las métricas de validación utilizadas para medir la efectividad del modelado predictivo, comenzando con la definición de las métricas utilizadas para evaluar el rendimiento de los modelos. Se han empleado métricas estándar conocidas, que permiten valorar con precisión la capacidad de los modelos para generalizar y ofrecer estimaciones fiables.

A continuación, se describen las técnicas de ML utilizadas para resolver dos tareas objetivo: la predicción del tiempo de estancia de los vehículos pesados en el recinto portuario y la estimación del volumen de tráfico por franjas horarias. Entre los enfoques explorados se incluyen modelos de regresión lineal, Random Forest y XGBoost, cada uno de ellos con sus propias ventajas en cuanto a precisión, interpretabilidad y eficiencia computacional.

Por último, se explica cómo los modelos seleccionados serán integrados en la aplicación final, centrandose en lo que respecta a los propios modelos predictivos y dejando clara su funcionalidad práctica dentro de la aplicación desarrollada. Así, este capítulo no solo justifica las decisiones técnicas tomadas, sino que también establece el vínculo entre la parte analítica del proyecto y su aplicación real.

6.1. Métricas de determinación y validación

Para evaluar la calidad de los modelos entrenados se emplearon distintas métricas, específicamente para las tareas de regresión realizadas en el marco de este trabajo. Las métricas seleccionadas fueron el coeficiente de determinación (R^2), el error absoluto promedio (MAE) y el error cuadrático medio (MSE), todas ellas ampliamente utilizadas en problemas de predicción de regresión con objetivos numéricos en competiciones de Kaggle.

En el caso particular del trabajo, en el que se estima la duración de estancia y el volumen de tráfico portuario, estas métricas ofrecen perspectivas complementarias sobre la calidad del modelo. El R^2 mide la proporción de la varianza explicada por el modelo respecto a los datos reales, lo que en este contexto permite evaluar qué tanto influye la información disponible (fecha, acceso, hora, etc) en la capacidad predictiva del sistema. Aunque su rango teórico va desde valores negativos hasta 1, en la práctica se busca que se aproxime lo más posible a 1, lo que reflejaría una capacidad explicativa adecuada.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Por su parte, el MAE representa el error promedio absoluto, permitiendo estimar de forma directa cuántas unidades (en horas o minutos o segundos) se desvía el modelo, en

promedio, respecto a los valores reales. Esta métrica es menos sensible a valores atípicos que otras, por lo que resulta especialmente útil cuando se quiere tener una estimación estable del error sin que pocos casos extremos distorsionen la media de error general.

$$\mathrm{MAE}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} |y_i - \hat{y}_i|}{N}$$

El MSE también calcula un error promedio, pero al elevar al cuadrado las diferencias, penaliza de forma mucho más severa las predicciones con grandes desviaciones. En este trabajo, esto resulta especialmente útil para identificar fallos puntuales pero graves en el rendimiento del modelo. Por ejemplo, es posible que el sistema obtenga buenos resultados generales, pero falle sistemáticamente al predecir los tiempos de estancia en una puerta concreta del puerto. Aunque ese fallo afectaría poco al MAE, sí tendría un impacto significativo en el MSE, lo que permite detectar y corregir sesgos o errores estructurales localizados en el modelo.

$$\mathrm{MSE}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{N}$$

La combinación de estas métricas permitió una evaluación completa del rendimiento de los modelos desarrollados, ya que cada una capturaba distintos aspectos del error. En conjunto, las métricas seleccionadas tienen interpretabilidad mediante R^2, precisión general a través del MAE, y sensibilidad ante errores significativos mediante el MSE. Este enfoque ayuda a construir modelos más equilibrados, capaces de mantener errores medios razonables, sin que se produzcan desviaciones especialmente graves en casos concretos, lo que aporta mayor robustez y consistencia a los resultados obtenidos.

Estas métricas fueron escogidas debido a su complementariedad: R^2 ofrece una visión global de la capacidad explicativa del modelo, el MAE aporta una estimación realista del error medio que puede esperarse y el MSE permite detectar si el modelo produce errores especialmente distantes del esperado. La combinación de estas proporciona un análisis relativamente completo para los modelos predictivos objetivo en este trabajo.

6.2. Modelos predictivos

En esta sección se presentan los modelos predictivos desarrollados para estimar la variable objetivo principal del proyecto: el tiempo de estancia de los vehículos en el recinto portuario. Dado que se trata de una variable continua, el problema se aborda como una tarea de regresión supervisada, siendo por tanto adecuados los modelos capaces de predecir valores numéricos con precisión.

La selección de modelos se basó inicialmente en recomendaciones, comenzando con enfoques clásicos como la regresión lineal y Random forest. Posteriormente se incorporó XGBoost, modelo ampliamente utilizado en competiciones en Kaggle por su rendimiento en tareas de regresión. Aunque no se realizó un estudio teórico profundo previo para cada algoritmo, la experimentación permitió identificar sus fortalezas y debilidades dentro de este trabajo.

6.2.1. Regresión lineal

La regresión lineal es un modelo predictivo, utilizado como punto de partida de modelos entrenados para regresión y series temporales, especializado en crear funciones algebraicas

que puedan representar futuros datos esperados. Este tiene como objetivo el establecer una relación funcional entre una o más variables independientes (entrada) y una variable dependiente continua (salida), lo que genera una función algebraica como:

$$y = a + bx_1 + cx_2 + \ldots + dx_n + \varepsilon$$

En la expresión mostrada,

$$y$$

representa la variable objetivo, como puede ser el tiempo de estancia de un camión en el puerto. Las variables x_n son las variables de entrada, como por ejemplo la puerta de acceso, el día de la semana o el momento del día. El término ε representa el error residual, mientras que los coeficientes $a, b, c, \ldots$ son los pesos ajustados durante el entrenamiento, que reflejan la influencia relativa de cada variable sobre la predicción final.

Por una parte, este modelo posee una serie de ventajas que resultan útiles para este trabajo. Destaca su mínimo coste computacional, una implementación sencilla sin necesidad de configuraciones complejas, y una interpretación clara de los resultados, lo que facilita la detección de posibles anomalías o patrones inesperados en los datos. Además, los coeficientes de valoración obtenidos permiten valorar directamente la influencia de las variables de entrenamiento, lo que convierte a la regresión lineal en una herramienta idónea para entrenamientos exploratorios.

Por otro, un modelo predictivo de regresión lineal tiene varias limitaciones destacables, pues debido a su relación entre variables estrictamente lineal tiene dificultades para capturar patrones complejos o no lineales. Una manera de expandir las posibilidades del modelo para crear funciones más complejas es codificar las variables complejas mediante métodos que introduzcan una expresividad en el modelo, esto se puede lograr con técnicas como la codificación cíclica o la polinómica. Asimismo, es sensible a la presencia de valores fuera de lo común, los cuales pueden afectar significativamente la estimación de los coeficientes y por consiguiente la precisión del modelo.

A pesar de dichas limitaciones, sirve como modelo base para el desarrollo futuro de otros modelos más complejos, permitiendo evaluar la dificultad del problema y calidad de los datos disponibles.

6.2.2. Random forest

Random forest [28] es una técnica basada en el uso de múltiples árboles de decisión, combinando las predicciones de estos árboles para obtener resultados sólidos. Este modelo predictivo pertenece a la familia de métodos ensamblados, donde varios estimadores simples colaboran para reducir el sobreajuste y mejorar la precisión resultante.

En tareas de regresión, como es el caso, este modelo predictivo construye dichos árboles a partir de subconjuntos de los datos y de las variables, generando un conjunto de pequeños modelos que se agregan utilizando el promedio. Cada árbol sigue su criterio de partición en función de la variable que más reduzca la impureza en cada nodo, permitiendo capturar relaciones no lineales entre variables.

En el contexto de este trabajo, Random forest resulta especialmente util debido a su capacidad para modelar de manera robusta la interacción entre variables de distinta naturaleza, como la puerta de acceso (categórica), o la hora (numérica). Estas relaciones no siempre son lineales ni independientes entre sí, por lo que una estrategia basada en árboles múltiples resulta útil para detectar patrones ocultos en los datos reales del entorno portuario.

Entre sus ventajas para el trabajo destaca su robustez frente a valores atípicos, su capacidad para generalizar sin necesidad de un ajuste fino de hiperparámetros y su tolerancia a datos ruidosos o parcialmente irrelevantes, algo habitual cuando se trabaja con registros operativos históricos. Gracias a su mecanismo de agregación aleatoria, se minimiza además el riesgo de sobreajuste típico en árboles individuales.

Sin embargo, este modelo también presenta algunas limitaciones. El entrenamiento puede resultar costoso computacionalmente, especialmente si se entrena con un gran número de estimadores o sobre conjuntos de datos extensos, como suele suceder en datos históricos. Además, el modelo final puede consumir más memoria y ser más lento en tareas de inferencia que otros algoritmos más ligeros.

A pesar de ello, el Random forest es un buen candidato como modelo final. Ofrece un equilibrio entre rendimiento, robustez y facilidad de uso, lo que lo convierte en una opción competitiva sin necesidad de una fase de ajuste de parámetros especialmente intensa.

6.2.3. XGBoost

El último modelo predictivo es XGBoost [29] (*Extreme Gradient Boosting*), una técnica basada en árboles popular por su buen rendimiento en tareas de regresión. XGBoost forma parte de la familia de algoritmos *boosting*, los cuales construyen modelos en secuencia, donde cada nuevo árbol intenta corregir los errores cometidos por los anteriores con el objetivo de obtener una mayor precisión. Como se puede determinar, este es un modelo ensamblado como Random forest. Con la diferencia en que los árboles son entrenados de manera independiente y se utiliza su promedio de resultados, ajustando los residuos del anterior árbol.

En el contexto del presente trabajo, XGBoost se exploró como modelo avanzado para la predicción de la duración de estancia de vehículos en el puerto. Este modelo es especialmente adecuado para capturar relaciones complejas entre variables categóricas y numéricas como la hora del día, la puerta de entrada, o el día de la semana. Su estructura secuencial permite detectar combinaciones no triviales entre factores, lo que resulta útil en un entorno con patrones temporales y operativos tan variados como el portuario.

Otra de las ventajas clave de XGBoost es su capacidad para manejar valores faltantes de forma automática, así como su sistema interno de regularización, que ayuda a evitar el sobreajuste sin necesidad de intervenciones manuales excesivas. Esto es particularmente útil cuando se trabaja con datos reales y operativos, donde la calidad de los registros puede variar, y donde se prefiere mantener la complejidad del *pipeline* de entrenamiento bajo control.

Sin embargo, este modelo también presenta limitaciones. La gran cantidad de hiperparámetros disponibles puede dificultar su configuración inicial, y en numerosos casos, la complejidad del modelo lleva a sobreajuste si no se aplican mecanismos adecuados de validación.

Dado su rendimiento competitivo y su capacidad para adaptarse a la complejidad del problema, XGBoost fue considerado como uno de los candidatos más prometedores en el proceso experimental. Sus resultados se compararon directamente con los obtenidos por modelos más simples, como la regresión lineal, y Random forest, para determinar su viabilidad como modelo de producción.

6.3. Resultados de los modelos predictivos estudiados

En esta sección se presentan los resultados obtenidos tras el entrenamiento y evaluación de los modelos predictivos desarrollados. Los modelos fueron entrenados sobre los conjuntos de datos preparados en las fases anteriores y validados utilizando una partición del 10 % de los datos reservada exclusivamente para prueba. Esta división garantiza una evaluación objetiva del rendimiento, evitando que los resultados estén condicionados por los datos de entrenamiento.

El análisis de resultados se estructura según los dos objetivos definidos en el proyecto: la predicción del *tiempo de estancia* y la estimación del *tráfico por franja horaria*. Dentro de cada uno de estos apartados, los algoritmos se presentan en el siguiente orden: regresión lineal, Random forest y XGBoost, siguiendo una progresión de complejidad.

6.3.1. Tiempo de estancia

Los resultados obtenidos por el modelo lineal entrenado son los siguientes: $R^2 = 0{,}1$, MSE = 73.485,62 y MAE = 183,22 minutos. De estos resultados se concluye que el modelo no parece ajustarse bien a los resultados, sólo un 10 % de la variación está siendo representada y los dos errores promedio presentan más de 3 horas de diferencia con respecto al valor esperado. Además, tiene un error cuadrático increíblemente alto, lo que significa que el modelo produce errores grandes. Como se puede observar en la figura 6.1, la distribución de los resultados del modelo por hora del día no captura las características de los datos, la variación entre horas parece ser bastante menor de la deseada, esto se debe a que otras características han tomado mayor importancia como son los accesos de entrada y salida. Aun así, se muestra una leve oscilación en la mediana de los valores estimados, reflejo del uso de codificación cíclica.

Por otra parte, la distribución de predicción muestra la existencia valores negativos, estos valores no tienen uso práctico y son el peor tipo de error posible para este trabajo pues no es posible que esto suceda en la realidad. Esto sucede debido a que en regresión lineal cuando se asignan los coeficientes a las variables, estos pueden ser positivos o negativos. La acumulación de coeficientes negativos generales produce valores que no presentaban en ningún momento los datos originales.

La codificación aplicada permite una mejor adaptación a los datos que un modelo puramente lineal, aunque sigue siendo insuficiente para capturar la complejidad real de los datos. Lo que llevo al desarrollo de otros modelos predictivos. Además, la regresión lineal asume relaciones lineales entre las variables independientes y la variable objetivo, lo que limita su capacidad para capturar patrones complejos o interacciones no lineales presentes en el comportamiento logístico real del puerto.

Los valores o pesos presentes en la regresión aplicada son los mostrados en la figura 6.2. En esta figura, se puede interpretar como se visualizó en secciones anteriores del capítulo la importancia del año, cuyo coeficiente es negativo indicando que el tiempo de estancia disminuye con el paso de los años, lo cual puede estar asociado a mejoras operativas o cambios en la gestión del puerto. Otras características como el día del mes experimentan patrones parecidos.

Por otro lado, el acceso de entrada muestra un coeficiente más alto que otras variables categóricas, lo que sugiere que el punto de acceso utilizado al entrar al puerto tiene una influencia notable sobre el tiempo de estancia. Esta característica aparece como la más

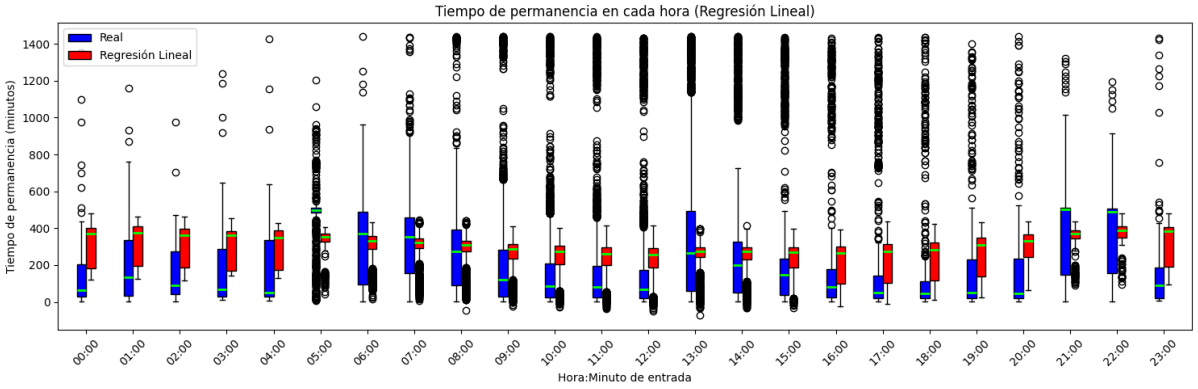

Figura 6.1: Distribución de los errores del modelo lineal por hora del día (tiempo estimado, minutos)

determinante, incluso más que los minutos del día, lo que refuerza la hipótesis de que la zona de entrada condiciona en gran medida la operativa posterior.

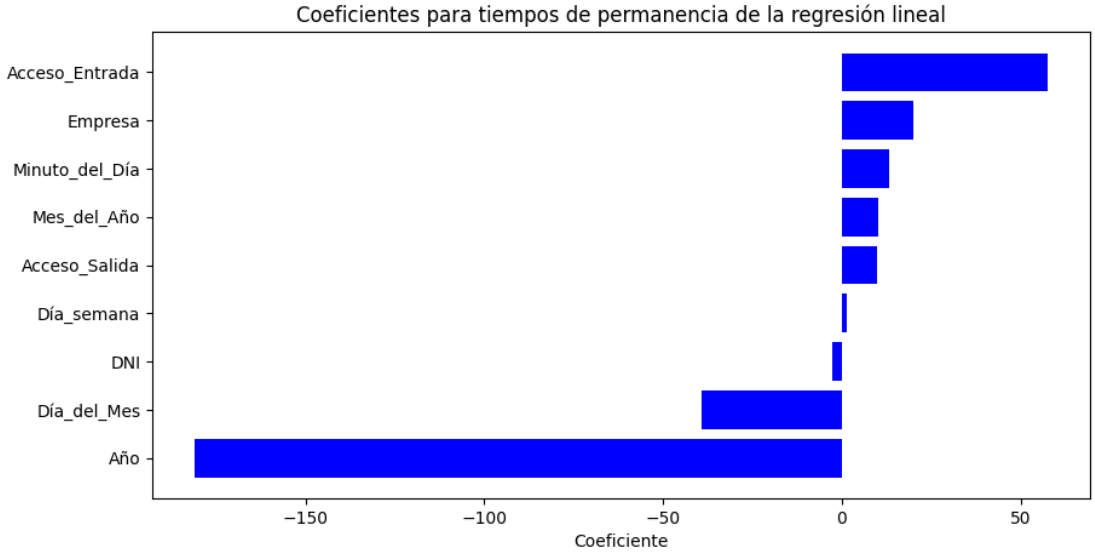

Figura 6.2: Distribución de coeficientes del modelo de regresión lineal para el tiempo estimado, minutos

El modelo entrenado de Random forest, configurado con 250 árboles de decisión, presenta una mejora notable respecto a la regresión lineal: $R^2 = 0{,}34$, MSE = 53.853,45 y MAE = 130,18 minutos. Estos valores indican que el modelo es capaz de capturar con mayor precisión la estructura subyacente de los datos, explicando un 34 % de la variabilidad en el tiempo de estancia, y reduciendo el error absoluto medio en más de 50 minutos con respecto al modelo anterior.

En los datos mostrados en 6.3, se visualizan cambios con respecto a la distribución del lineal 6.1. Aunque este, sigue produciendo errores en las horas se destaca la mejora de alrededor de un 20 % en comparación a la regresión lineal y la exclusión de resultados negativos lo que, aunque continúa siendo impreciso resulta en un modelo utilizable para ser utilizado en las predicciones. Los resultados muestran por lo menos la tendencia apropiada, aun así debido a la gran cantidad de muestras en los datos reales que no aparentan seguir

un patrón reconocible, el resultado final no se espera que pueda mejorar demasiado.

La desventaja principal de este modelo termina siendo el tiempo que se tardan en procesar los datos desde que se inicia el script hasta que se devuelve un resultado, pues se tarda alrededor de 17 segundos en cargar el modelo entrenado y otros 5 segundos en procesar y codificar los datos. Tardando un mínimo de 17 segundos antes de poder devolver un simple resultado se considera demasiado.

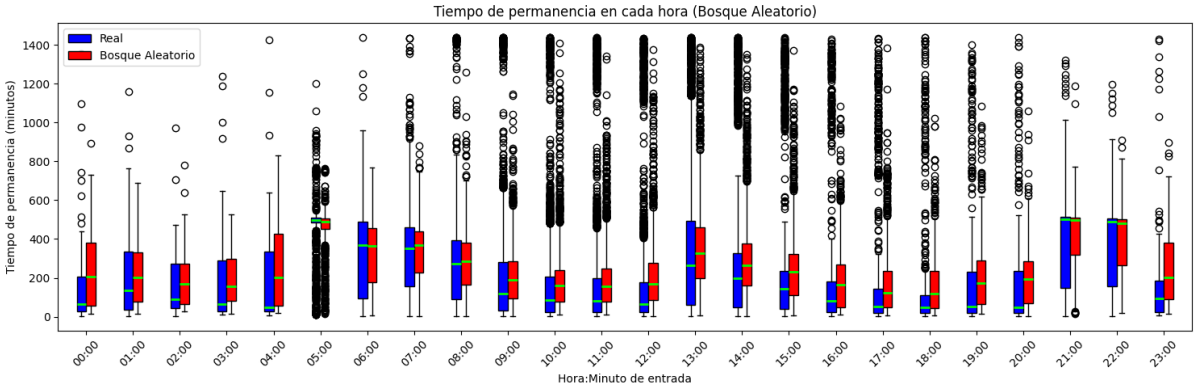

Figura 6.3: Diagrama de barras con la distribución de resultados del modelo de Random forest por hora del día para el tiempo estimado, minutos

El modelo utilizando 500 estimadores de XGBoost logró resultados similares al Random forest, con una ligera mejora en variación y error medio cuadrático pero conllevo una pequeña perdida en error medio absoluto: $R^2 = 0{,}35$, MSE = 53.398,14 y MAE = 138,13 minutos. Aunque el MSE mejora, el MAE empeora lo que sugiere una mejor gestión de los errores grandes, mientras que errores más leves se vuelven más comunes.

La figura 6.4 muestra la distribución horaria de los resultados. Se aprecia que el modelo logra seguir razonablemente la tendencia general del tiempo de estancia por hora del día, aunque con menor fidelidad en las horas de mayor varianza. Sin embargo, uno de los principales problemas identificados es la generación de valores negativos. Aunque aparecen con menor frecuencia que en la regresión lineal, siguen estando presentes, lo cual limita la aplicabilidad del modelo, ya que este tipo de predicción carece de sentido práctico en el contexto del problema.

A pesar de esta limitación, XGBoost ofrece una ventaja operativa clara en cuanto al rendimiento computacional: el modelo se carga en aproximadamente 0,5 segundos, y la predicción sobre nuevos datos apenas requiere 0,2 segundos adicionales. Esta eficiencia lo convierte en una alternativa viable para entornos donde la velocidad de inferencia sea un factor relevante como es el caso, pues no se puede tener a un usuario esperando un cuarto de minuto cada vez que quiera utilizar estos modelos.

Tabla 6.1: Tabla comparativa de modelos para la predicción del tiempo de estancia

Modelo	**MAE**	**MSE**	**R^2**
Regresión lineal	183,22	73.485,62	0,1
Random forest	130,18	53.853,45	0,34
XGBoost	138,13	53.398,14	0,35

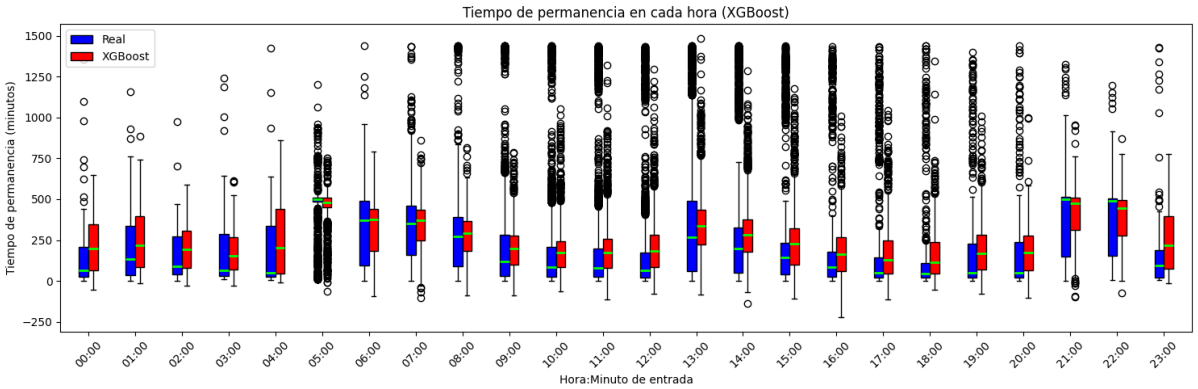

Figura 6.4: Diagrama de barras con la distribución de resultados del modelo de XGBoost por hora del día para el tiempo estimado, minutos

Tabla 6.2: Tabla comparativa de modelos para el tiempo de computo del tiempo de estancia

Modelo	**Tiempo de carga aproximado (segundos)**
Regresión lineal	0,1
Random forest	17
XGBoost	0,2

6.3.2. Tráfico

La regresión lineal para la predicción de tráfico obtuvo: $R^2 = 0{,}29$, MSE = 195,19 y MAE = 8,67 accesos. Los resultados demuestran que una proporción significativa de patrones están siendo capturados por el modelo. El coeficiente de determinación ($R^2 = 0{,}29$) indica que, si bien se logra cierta capacidad explicativa, aún queda una parte considerable de la variabilidad sin modelar, lo que limita su efectividad para entornos que requieran mayor precisión.

Aunque supera la precisión obtenida en la tarea de tiempo de estancia, sigue siendo insuficiente para tareas más exigentes. La figura 6.5 muestra una alta dispersión en los errores y evidencia la presencia de predicciones con valores negativos. Se detectaron predicciones de tráfico con valores negativos, lo cual no es realista en un contexto operativo, ya que el número de accesos no puede ser inferior a cero. Este fenómeno se observa principalmente en las primeras horas punta del día (5:00–8:00) y durante la tarde-noche (15:00–21:00), lo que reduce la confiabilidad del modelo en esas franjas.

A diferencia de los modelos predictivos de tiempo estimado, el tráfico utiliza pocas variables originales lo que facilita conocer la importancia de las características. En la figura se distingue la importancia de los accesos en comparación con las características temporales, pues esta segunda es de menor importancia para saber cuan de congestionada estará una puerta. Corroborando los resultados de que las puertas son de vital importancia.

A diferencia de los modelos predictivos de tiempo estimado, el modelo de predicción de tráfico utiliza menos variables, lo que facilita la interpretación de los pesos y la evaluación de la importancia de cada característica. En la figura 6.6, se aprecia claramente la influencia de los accesos frente a las variables temporales. Esta diferencia sugiere que

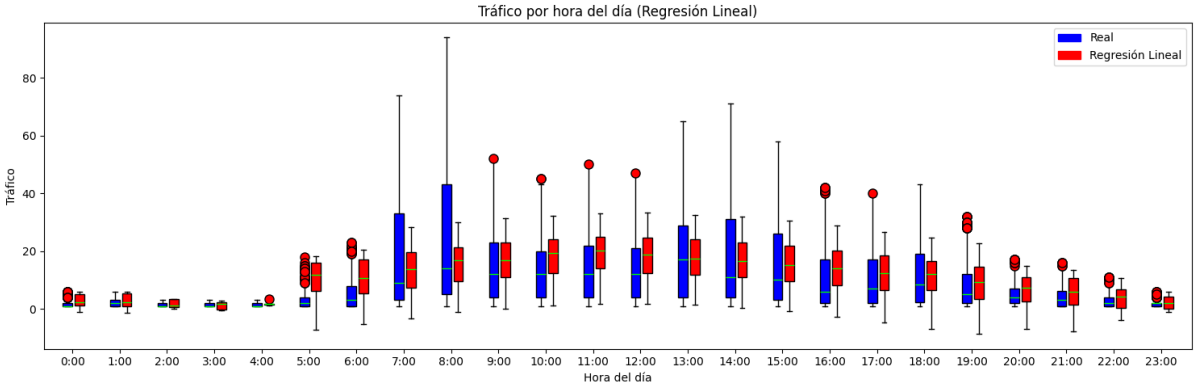

Figura 6.5: Diagrama de *boxplot* con la distribución de resultados del modelo de regresión lineal por hora del día para el tráfico

la puerta por la que se produce el acceso tiene un peso significativamente mayor a la hora de estimar el volumen de tráfico, mientras que la información temporal tiene un papel secundario.

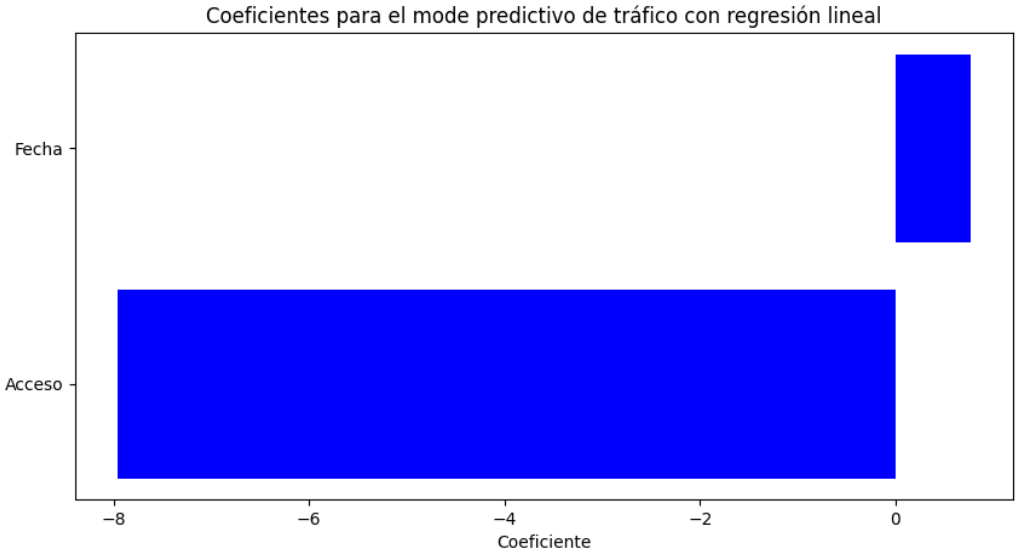

Figura 6.6: Diagrama de *boxplot* con los coeficientes del modelo de regresión lineal para el tráfico

El modelo entrenado de Random forest mejora significativamente con respecto a la regresión lineal: $R^2 = 0{,}79$, MSE = 57,23 y MAE = 4,06 accesos. Esta mejora supone prácticamente duplicar la precisión del modelo anterior: los errores medios se han reducido a más de la mitad, y el coeficiente de determinación se acerca al 80 %, lo que indica que el modelo es capaz de explicar la mayor parte de la variabilidad del tráfico horario.

Para alcanzar estos resultados, se ajustaron los hiperparámetros del modelo como la profundidad máxima de los árboles y la cantidad mínima de muestras por hoja con el objetivo de limitar el sobreajuste y mejorar la capacidad de generalización. El resultado es un modelo con un comportamiento más robusto y una mayor fidelidad a las dinámicas del tráfico portuario.

La figura 6.7 muestra cómo la distribución de las predicciones por hora del día se

ajusta adecuadamente a los datos reales. La mediana estimada por el modelo se alinea con la mediana real en la mayoría de los casos, reflejando un buen desempeño general. No obstante, se observan ciertas limitaciones en la gestión de valores atípicos o desviaciones extremas dentro de una misma franja horaria, donde el modelo tiende a suavizar el comportamiento y no capturar completamente la varianza real.

En cuanto al rendimiento computacional, el tiempo de carga del modelo es de aproximadamente 1,5 segundos, mientras que la generación de predicciones se ejecuta en 0,72 segundos. Estos tiempos resultan aceptables para su uso en la aplicación con restricciones.

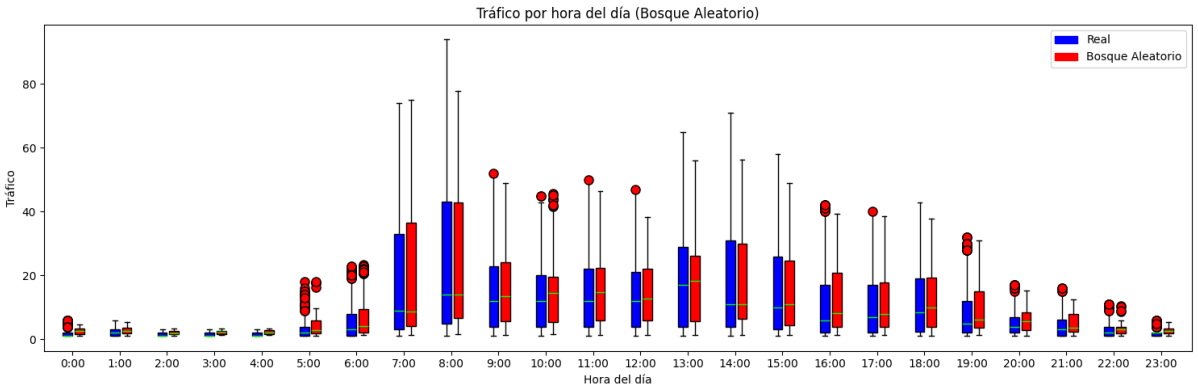

Figura 6.7: Diagrama de *boxplot* con la distribución de resultados del modelo de Random forest por hora del día para el tráfico

El modelo de XGBoost obtuvo el mejor rendimiento con diferencia en esta tarea: R^2 = 0,88, MSE = 34,35 y MAE = 3,7 accesos. Este resultado confirma que el modelo es capaz de capturar las dinámicas complejas de tráfico con elevada precisión. Aún asi en la distribución se observan problemas 6.8. Concretamente, una proporción de las predicciones presenta valores negativos, los cuales carecen de sentido práctico en este contexto, ya que el número de accesos no puede ser inferior a cero.

Este tipo de error es especialmente problemático porque, aunque poco frecuente, afecta gravemente a la validez de las predicciones. Esto complica la elección del modelo más adecuado: XGBoost comete menos errores en términos absolutos, pero los errores que sí produce pueden ser más graves que los de modelos ligeramente menos precisos como el Random forest, que en cambio evita predicciones absurdas.

Para alcanzar estos resultados, fue necesario ajustar gran parte de los hiperparámetros por defecto del modelo. Entre los más relevantes destacan: una tasa de aprendizaje de aproximadamente 0,16, una profundidad máxima de 9 niveles por árbol y un total de 850 estimadores. Estos ajustes permitieron optimizar el equilibrio entre precisión y sobreajuste.

Por otra parte, el rendimiento computacional como sucedio en el tiempo de estancia es bastante competitivo, pues el tiempo que tarda en estar listo el modelo es de 0,26 segundos mientras que los segundos que se tardan en procesar los datos de validación son 0,05 segundos.

En resumen, los modelos de árbol (Random forest y XGBoost) se mostraron más adecuados para ambas tareas como se observa en las tablas 6.1 y 6.3. Mientras que la regresión lineal sirve como punto de partida, no resulta competitiva cuando los patrones a aprender incluyen relaciones no lineales entre variables categóricas y temporales codificadas. Los tiempos de carga entre los árboles terminan siendo un factor clave a tener en

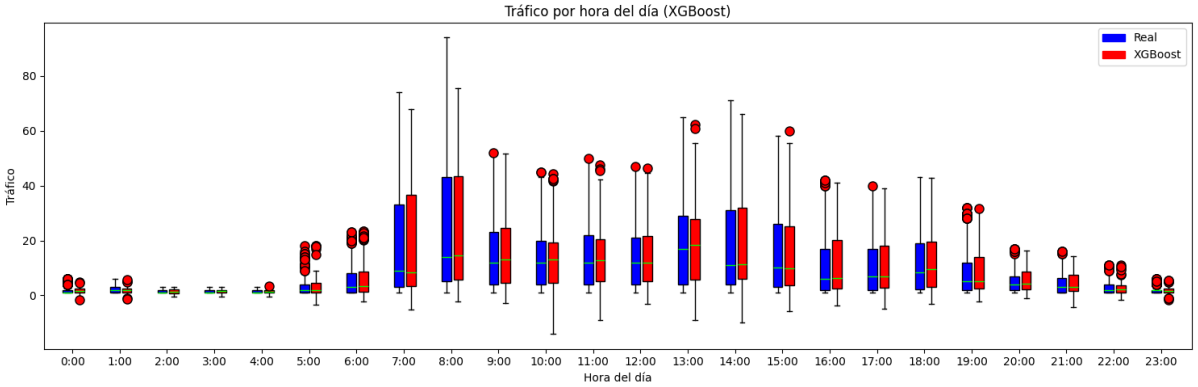

Figura 6.8: Diagrama de *boxplot* con la distribución de resultados del modelo de regresión lineal por hora del día para el tráfico

Tabla 6.3: Comparativa de modelos para la predicción del tráfico

Modelo	**MAE**	**RMSE**	**R^2**
Regresión lineal	8,67	195,19	0,29
Random forest	4,06	57,23	0,79
XGBoost	3,7	34,35	0,88

Tabla 6.4: Comparativa de modelos para el tiempo de computo del tráfico

Modelo	**Tiempo de carga (segundos)**
Regresión lineal	0,1
Random forest	1,5
XGBoost	0,2

cuenta, tablas comparativas de estos tiempos son 6.2, 6.4 los cuales resultan relevantes en el factor de decisión sobre el mejor modelo, puesto que si se necesitan realizar varias solicitudes se puede tardar demasiado.

Por estas razones, los modelos seleccionados para su integración en la aplicación final fueron los de XGBoost. En el caso del tiempo de estancia, se optó por XGBoost principalmente debido al elevado tiempo de carga del modelo Random forest, pese a que este último produce resultados más consistentes al evitar predicciones negativas. Si en el futuro se optimizara el modelo de Random forest en términos de carga, podría considerarse su sustitución. Para el modelo de tráfico, la decisión fue más clara: XGBoost ofrece una mayor precisión general y una ejecución más rápida. Los pocos casos en que produce valores negativos serán gestionados directamente en la aplicación, reemplazándolos por cero, dado que dichos valores carecen de sentido práctico.

6.4. Integración de los modelos entrenados en la aplicación

El conjunto de técnicas de *machine learning* desarrolladas, las cuales han sido empaquetadas en un *pipeline* conteniendo los modelos predictivos entrenados y las transforma-

ciones necesarias. Son exportados utilizando cloudpickle, permitiendo utilizar los modelos entrenados. A través de un script escrito en Python y un fichero con los datos proporcionados por la aplicación, el script obtendrá los datos de la capa de negocio y estos se los transmite a la tubería conteniendo al modelo predictivo correspondiente, finalmente el script devolverá un resultado útil para la aplicación.

El resultado de estas operaciones tiene como objetivo ser utilizado por la aplicación durante el caso de uso (Crear una reserva) para crear las gráficas que sirven de ayuda a los transportistas en la creación de las reservas con el objetivo de que estos elijan una hora de entrada o salida que les sea conveniente basándose en los datos mostrados.

Las figuras 6.9 y 6.10 muestran concretamente el uso de los resultados de los modelos dentro de la propia aplicación a modo de ejemplo en el caso de uso especificado. Estas predicciones mostradas en forma de gráficos de barras tienen el objetivo de servir a los transportistas información util para a beneficio tanto del propio puerto como de ellos realizar reservas cuando mejor sea para ambas partes.

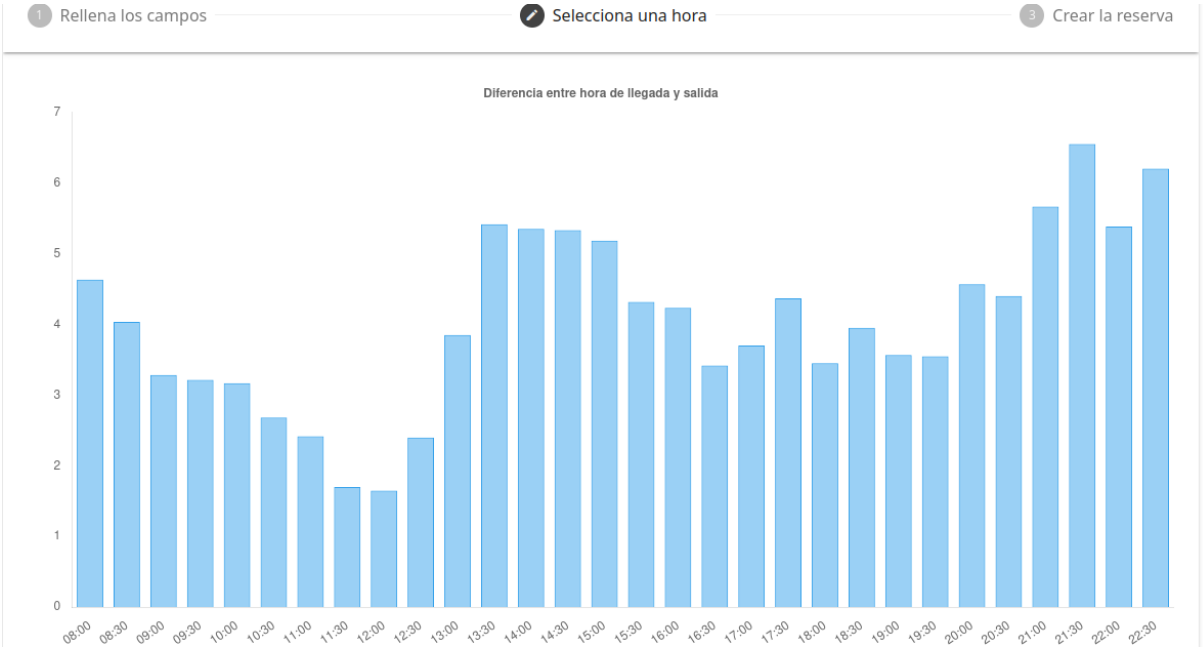

Figura 6.9: Gráfica presente en la aplicación creada con el modelo de tiempo estimado

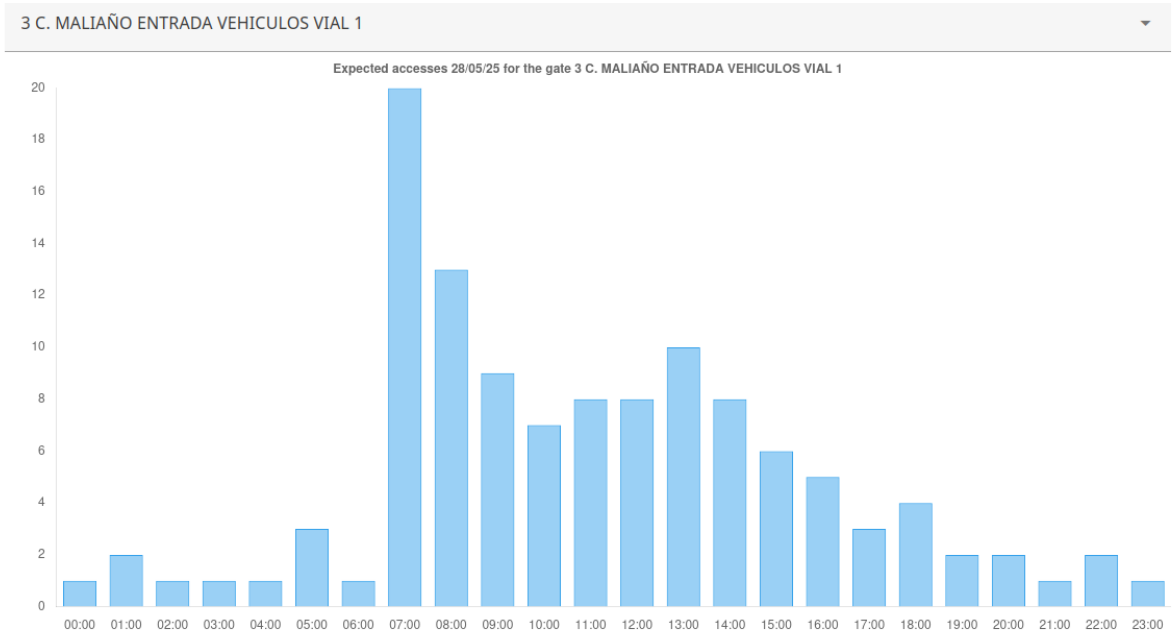

Figura 6.10: Gráfica presente en la aplicación creada con el modelo de tráfico

Capítulo 7

Conclusiones y líneas de trabajo futuro

El desarrollo de este trabajo ha dado lugar a una solución software completa orientada a la mejora del tráfico terrestre en recintos portuarios. A través de una aplicación full-stack y una serie de modelos predictivos integrados y entrenados, se ha logrado construir un sistema funcional que permite tanto la gestión operativa de accesos como la estimación anticipada del comportamiento de los vehículos pesados en el puerto. Esta solución se ha planteado desde el inicio como una herramienta práctica y ajustada a problemas reales observados en el Puerto de Santander.

7.1. Retos y objetivos

En relación con los objetivos propuestos, se puede afirmar que todos ellos han sido abordados de forma satisfactoria. Se ha desarrollado una aplicación web con diferentes perfiles de usuario, capaz de mostrar información histórica y resultados predictivos sin requerir conocimientos técnicos por parte del usuario final. También se han explorado distintas técnicas de aprendizaje automático, entrenado modelos sobre datos reales y evaluado su comportamiento mediante métricas objetivas. Los resultados obtenidos, aunque condicionados por la calidad de los datos disponibles, muestran un rendimiento aceptable en la tarea de predicción de tiempos de estancia y tráfico horario.

Uno de los principales retos encontrados durante el trabajo ha estado directamente relacionado con la calidad de los datos históricos. La ausencia de identificadores únicos para vincular las entradas y salidas de los vehículos obligó a realizar un proceso manual e impreciso de emparejamiento, además de una limpieza intensiva para garantizar la coherencia de las muestras utilizadas en el modelado. A esto se suma la falta de información relevante sobre el propósito del acceso, lo que limita la precisión máxima alcanzable por los modelos. Adicionalmente, los registros de acceso no incluyen un dato fundamental para la aplicación: la matrícula del vehículo. De haberse dispuesto de este dato, no solo se habrían obtenido modelos más precisos, sino también un conjunto más amplio y realista de camiones distintos. Esta carencia obligó a utilizar en todos los registros un camión genérico de relleno, ya que dicho atributo es esencial para construir un acceso válido en la aplicación. Se tomó la decisión de considerar el camión como un elemento obligatorio en la creación de un acceso, dado que es un dato que puede obtenerse fácilmente mediante lectores de matrículas automáticos, tecnología que ya se emplea en entornos urbanos como el de la Policía Local en Gran Canaria para identificar vehículos sin necesidad de

intervención humana.

La elección de diseño de la arquitectura hexagonal en la aplicación ha resultado especialmente ventajosa. Esta elección ha facilitado la integración de componentes como Keycloak para la autenticación, RabbitMQ para el sistema de mensajería y bases de datos especializadas como MongoDB y TimescaleDB. Esta modularidad permite que la aplicación pueda ser extendida o adaptada a distintos entornos sin necesidad de rediseñar su estructura interna, favoreciendo así su escalabilidad y mantenimiento.

Aunque no era un objetivo explícito al inicio del proyecto, durante el desarrollo también se incorporaron una serie de mejoras en la experiencia de usuario y en el diseño de la interfaz que refuerzan la usabilidad general de la herramienta. Entre ellas, destaca la inclusión de soporte multilingüe en español e inglés, lo que amplía la accesibilidad de la aplicación a distintos perfiles de usuario en contextos internacionales. También, se ha cuidado la coherencia visual, pues los colores, estilos y componentes se han mantenido consistentes entre las diferentes vistas, siguiendo una estética clara que facilita la navegación. Estas decisiones, aunque secundarias, refuerzan el valor práctico del software y su potencial como producto.

Por otro lado, en lo referente a la representación gráfica, se optó por el uso de elementos visuales simples como gráficas de barras, histogramas y *boxplots*. Aunque se trata de representaciones básicas, estas han resultado suficientes para mostrar de forma clara la información relevante extraída de los modelos y los datos históricos. Dada la naturaleza exploratoria del proyecto, se priorizó la claridad y la rapidez de interpretación frente a soluciones más complejas que pudieran resultar complicadas frente a usuarios con poca experiencia tratando datos.

Desde el punto de vista del modelado, se ha demostrado que enfoques como Random forest y XGBoost ofrecen mejores resultados que los modelos lineales clásicos en problemas como el de este trabajo, aunque a cambio de una mayor complejidad y coste computacional. Los análisis realizados también han puesto de manifiesto que los errores están correlacionados con franjas horarias específicas y que la distribución desigual de los datos de entrenamiento afecta directamente al rendimiento de los modelos. Esto refuerza la importancia de trabajar con conjuntos de datos equilibrados y ricos en variables contextuales para lograr modelos más precisos para el puerto en cuestión.

Además del cumplimiento de los objetivos técnicos, este trabajo ha demostrado la viabilidad de aplicar técnicas de inteligencia artificial en la logística portuaria desde una perspectiva aplicada. La herramienta desarrollada, si bien no es un producto final completamente operativo, constituye un paso importante hacia una solución que podría tener impacto real en entornos portuarios como el del Puerto de Santander. El sistema no solo permite gestionar e interpretar los datos históricos de forma visual e intuitiva, sino que también introduce un componente predictivo que puede anticipar congestiones y contribuir a una mejor toma de decisiones por parte de gestores y operadores.

7.2. Líneas de trabajo futuro

El desarrollo de esta aplicación ha permitido consolidar y ampliar conocimientos fundamentales en múltiples áreas clave de la ingeniería informática. A nivel técnico, se ha adquirido experiencia práctica en la creación de una solución web *full-stack*, integrando tecnologías tanto del lado cliente como del servidor, y estructurando la lógica de negocio en una arquitectura multicapa robusta y modular.

Uno de los mayores aprendizajes ha sido el trabajo con bases de datos no relacionales,

en concreto MongoDB, lo que ha supuesto una expansión significativa respecto al conocimiento previo centrado en bases de datos relacionales. Esto ha permitido comprender mejor los distintos enfoques de modelado y consulta de datos, y cuándo es más conveniente utilizar un tipo de bases de datos.

En cuanto al ámbito de la ciencia de datos, este trabajo ha supuesto una primera toma de contacto con plataformas como Kaggle, la cual ha sido de gran útilidad para aprender sobre este campo, dado que no forma parte habitual del plan de estudios del grado. Esta exploración ha demostrado que el campo del aprendizaje automático es un área con un altísimo potencial para generar aplicaciones útiles y con impacto real.

Aunque se consideraron modelos más avanzados como redes neuronales, estos fueron finalmente descartados debido a su complejidad, menor rendimiento con los datos disponibles y mayores requisitos computacionales. Este proceso de evaluación comparativa también ha sido formativo, permitiendo desarrollar criterio técnico para elegir la mejor herramienta según el contexto.

Por último, el uso de herramientas como Keycloak para la autenticación de usuarios, RabbitMQ para la mensajería entre servicios, han contribuido a desarrollar una visión más completa y realista de cómo diseñar e implementar una solución software moderna. En conjunto, este trabajo ha reforzado no solo habilidades técnicas, sino también una forma de pensar orientada a la integración, reutilización y toma de decisiones fundamentadas, sentando una base sólida para futuros proyectos en entornos profesionales o de investigación.

De cara al futuro, se identifican varias líneas de trabajo que permitirían consolidar y ampliar las funcionalidades de la aplicación. En primer lugar, sería conveniente incorporar nuevos conjuntos de datos que aporten mayor riqueza y contexto. Variables como el tipo de carga transportada, el propósito del acceso al recinto, el peso del vehículo o incluso la meteorología podrían influir de manera significativa en los tiempos de estancia, y su inclusión permitiría desarrollar modelos predictivos más robustos y precisos.

Una mejora relevante en el ámbito del rendimiento tiene que ver con la forma en que se accede a los datos. En el estado actual, la aplicación no utiliza paginación en las consultas a las bases de datos excepto por la entidad de accesos, lo que puede afectar negativamente a su eficiencia a medida que crece el volumen de información. Incorporar mecanismos de paginación permitiría optimizar la carga de datos y reducir el impacto en el tiempo de respuesta. Además, en tareas más exigentes como la importación masiva de registros, sería posible aprovechar RabbitMQ para delegar el procesamiento a un sistema asíncrono. Aunque RabbitMQ ya está integrado, haría falta configurar nuevas colas y adaptadores específicos para este propósito. Esto permitiría realizar operaciones intensivas sin bloquear la interfaz principal ni afectar a la experiencia del usuario.

Otra mejora clave sería una integración más completa del sistema de autenticación con Keycloak. Actualmente, se dispone de roles generales, pero extender esta funcionalidad para vincular transportistas concretos a cuentas de usuario específicas permitiría personalizar las vistas de los datos. Esto, por ejemplo, facilitaría que un conductor pudiera ver únicamente sus propios registros de acceso y predicciones asociadas, reforzando la utilidad del sistema desde una perspectiva práctica y de privacidad. Lo que permitiría también que los conductores pudieran acceder a vistas actualmente restringidas para ellos como la lista de accesos.

Por otra parte, una continuación lógica del proyecto sería la integración de la planificación de buques en la aplicación. Unificar la gestión del tráfico terrestre con la información sobre atraques, llegadas y operaciones marítimas permitiría ofrecer una visión completa

del entorno portuario. Esta fusión de información contribuiría a una coordinación más eficiente entre la logística marítima y terrestre, alineada con los principios de los puertos inteligentes y la logística 4.0. Esta continuación, aunque pesada, pues conllevaría grandes cambios en la aplicación, resultaría en una aplicación con una utilidad potencial para la entidad portuaria mucho mayor.

Por último, un paso fundamental sería validar la herramienta en un entorno real de producción, con usuarios operativos del puerto. Este tipo de despliegue permitiría no solo identificar mejoras funcionales adicionales, sino también evaluar el impacto directo de la aplicación al completo en la eficiencia logística, la reducción de esperas y la optimización del uso de infraestructuras portuarias.

En conclusión, este trabajo representa una aproximación práctica y funcional a la problemática de la congestión terrestre de los camiones en puertos. A través de la combinación de desarrollo software y técnicas de aprendizaje automático, se ha construido una herramienta que permite no solo gestionar y visualizar datos, sino también anticipar situaciones de congestión y mejorar la toma de decisiones. Si bien existen limitaciones técnicas derivadas del acceso a los datos y la fase temprana del desarrollo, la aplicación desarrollada sienta una buena base sobre la que se pueden construir soluciones más completas, escalables y alineadas con los desafíos de la logística portuaria actual.

Apéndice A

Competencias específicas cubiertas

Durante la elaboración de este trabajo de fin de grado, se han aplicado diversas competencias específicas relacionadas con la ingeniería informática, estas son:

A.1. Gestión de proyectos y administración de sistemas (CI2, CI5)

La competencia CI2 se materializó en la planificación y despliegue integral del proyecto, desde su creación hasta su ampliación como TFG, siguiendo un ciclo de vida iterativo incremental durante el transcurso, sobre todo del trabajo. Por su parte, CI5 se aplicó en la administración de la infraestructura técnica relacionada con los contenedores, los cuales utilizan tecnología Docker: despliegue de contenedores, mantenimiento de las bases de datos y gestión de las dependencias de cada módulo.

A.2. Diseño de estructuras y arquitectura de software (CI7, CI8)

El diseño de clases pertenecientes al dominio como Reserva, Acceso y Puerto aseguro la aplicación de la competencia CI7, asegurando estructuras de datos eficientes para ser utilizadas por las operaciones que sean. CI8 se aplicó al adoptar una arquitectura hexagonal que separa lógica de negocio (casos de uso), de la interfaz e infraestructura, garantizando modularidad y facilitando la integración, nuevas funcionalidades o el uso de otros modelos de aprendizaje automático.

A.3. Gestión y procesamiento de datos (CI12, CI13)

CI12 se reflejó en el diseño del esquema relacional de la aplicación (tablas para los accesos), se definió como se guardaría en el esquema de la base de datos la clase, asegurando las claves primarias, lo que permite búsquedas eficientes en la base de datos. CI13 cubrió el procesamiento de datos históricos (ej: tiempos de estancia en el Puerto de Santander) y su combinación con variables en tiempo real (tráfico, mareas), esencial para generar recomendaciones proactivas.

A.4. Sistemas inteligentes y web (CI15, TI6, CI17)

CI15 ha sido clave en el desarrollo de modelos de ML para predecir los tiempos de estancia y el tráfico, usando las bibliotecas Scikit-learn y XGBoost. Finalmente, el valor de TI6 se aplicó al construir una aplicación full-stack basada en el estilo web, con servicios REST para comunicación cliente-servidor y la renderización dinámica de las interfaces utilizando herramientas como Vue.js. La competencia CI17 fue aplicada para una experiencia de usuario intuitiva: gráficos legibles para gestores (gráficos de accesos) y vistas simplificadas para conductores (reservas con recomendaciones horarias a través de una serie de pasos).

Apéndice B

Objetivos de desarrollo sostenible

Los ODS, cuya lista completa se puede ver en la tabla B.1, presentan distintos niveles de relación con este trabajo. De entre ellos, los más vinculados se justifican por las razones siguientes:

El sistema desarrollado durante el trabajo está orientado a la mejora de las infraestructuras portuarias mediante técnicas en innovación constante como él *machine learning*. Estas tecnologías en la aplicación se utilizan con el objetivo de lograr una gestión más eficiente del tráfico portuario y de los recursos logísticos, dando lugar a la innovación en los sectores de transporte y logística.

Una mejor planificación del acceso y del flujo de vehículos en zonas portuarias puede contribuir en la reducción del impacto ambiental por vehículos pesados. Permitiendo que los puertos sean más limpios en gases, resultando en una ciudad más limpia y la congestión portuaria no genere tráfico en las carreteras generales de la ciudad.

Los modelos predictivos contribuyen indirectamente a las rutas elegidas por los transportistas de camiones y un uso más eficiente de los recursos logísticos. Estas mejoras reducen el consumo innecesario de energías fósiles de los vehículos y el desperdicio de recursos operativos, alineándose con el objetivo de promover prácticas de producción y consumo sostenibles.

ODS	**Nivel de relación**			
	0	**1**	**2**	**3**
1. Fin de la Pobreza	**X**			
2. Hambre cero	**X**			
3. Salud y Bienestar		**X**		
4. Educación de calidad	**X**			
5. Igualdad de género	**X**			
6. Agua limpia y saneamiento	**X**			
7. Energía asequible y no contaminante	**X**			
8. Trabajo decente y crecimiento económico		**X**		
9. Industria, Innovación e Infraestructuras				**X**
10. Reducción de las desigualdades	**X**			
11. Ciudades y comunidades sostenibles				**X**
12. Producción y consumo sostenibles			**X**	
13. Acción por el clima		**X**		
14. Vida submarina	**X**			
15. Vida de ecosistemas terrestres	**X**			
16. Paz, justicia e instituciones sólidas	**X**			
17. Alianzas para lograr objetivos	**X**			

Tabla B.1: Tabla con la valoración del grado de relación del TFG con los ODS, según una escala de 0 a 3 (no procede, bajo, medio, alto).

Apéndice C

Manual de usuario

Este apéndice, Está dedicado a mostrar las vistas del *front-end* y a servir como guía de como utilizar la aplicación para uno de los usuarios que la utilicen. Se asume el conocimiento previo de los requisitos presentes en la aplicación a desarrollar.

C.1. Inicio de sesión

Como se menciona en el capítulo de desarrollo 4, para poder utilizar la aplicación, un usuario necesita estar registrado. Para poder realizar distintas pruebas fueron creados 3 usuarios, uno por cada rol de la aplicación: **rol(usuario, contraseña)**

Administrador(administrador, 123), Conductor de camión(paco, 123), Gestor portuario(peter, 123).

Una vez el usuario haya iniciado su sesión, quedará ubicado en la página de inicio C.1. Seleccionando el icono de la bandera se puede cambiar de idioma.

C.2. Navegación y notificaciones

Para acceder a las páginas disponibles para el rol actual del usuario con el que se haya iniciado sesión, se ha de clicar en el icono de hamburguesa en la parte superior izquierda de la pantalla. Dentro del desplegable abierto tras realizar la operación anterior se tienen dentro las páginas accesibles distribuidas por entidad (puerto, camión ...), la figura muestra la vista de navegación del administrador C.2. Se puede acceder a una página de una entidad clicando en la entidad correspondiente, lo que abre un panel en el que se podrá seleccionar la funcionalidad deseada.

Existen vistas particulares que redirigen a otras vistas de la aplicación, como cuando se finaliza la creación de una reserva, el usuario será redirigido a la vista de reservas en donde podrá identificar su reserva recién creada.

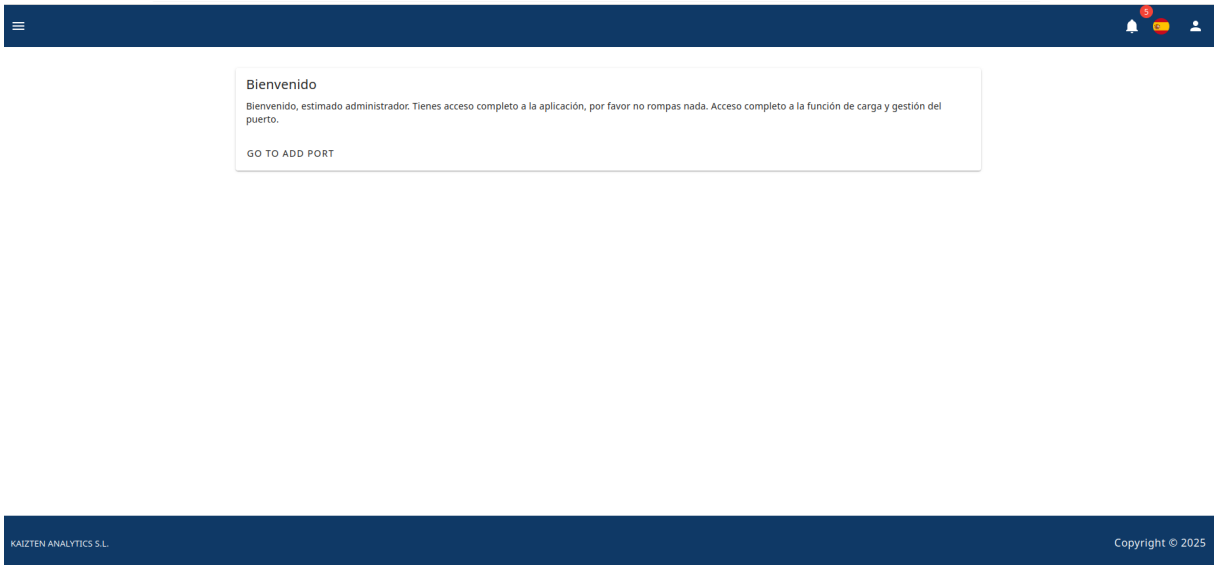

Figura C.1: Página de inicio

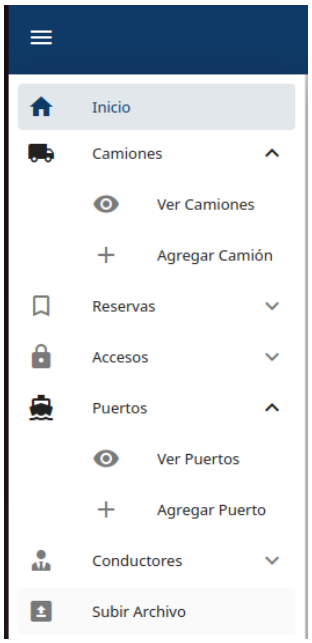

Figura C.2: Columna de navegación de la aplicación

Cuando sucedan cambios en los datos de la aplicación, los usuarios reciben notificaciones como las mostradas en C.3, permitiéndoles conocer cambios que pudieran afectar a todos los usuarios. Aunque no se especifica el usuario que realizó dicho cambio.

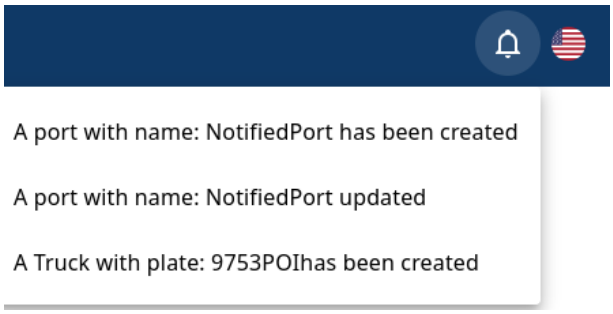

Figura C.3: Notificaciones de la aplicación

C.3. Manejo de entidades

Desde cualquier entidad disponible en el panel de navegación, y dependiendo de los permisos del usuario, se puede acceder a una página de tipo Ver (como Ver puertos), identificada con un icono en forma de ojo. Estas vistas, comunes a las distintas entidades, permiten visualizar de forma resumida las instancias registradas.

Para mostrar los datos, se emplea una tabla como las que se observan en las figuras C.4 y C.5. Estas tablas comparten algunas características: el texto en azul actúa como enlace a la vista de detalles de la instancia seleccionada (excepto en el caso de los accesos, cuya vista de detalle no fue implementada), el icono de engranaje permite editar la instancia correspondiente a la fila y la cruz roja permite eliminarla.

Finalmente, estas vistas incorporan un buscador que permite filtrar y localizar una instancia específica dentro de la tabla. No obstante, en el caso de los accesos, esta funcionalidad no es compatible con la paginación utilizada por esa entidad.

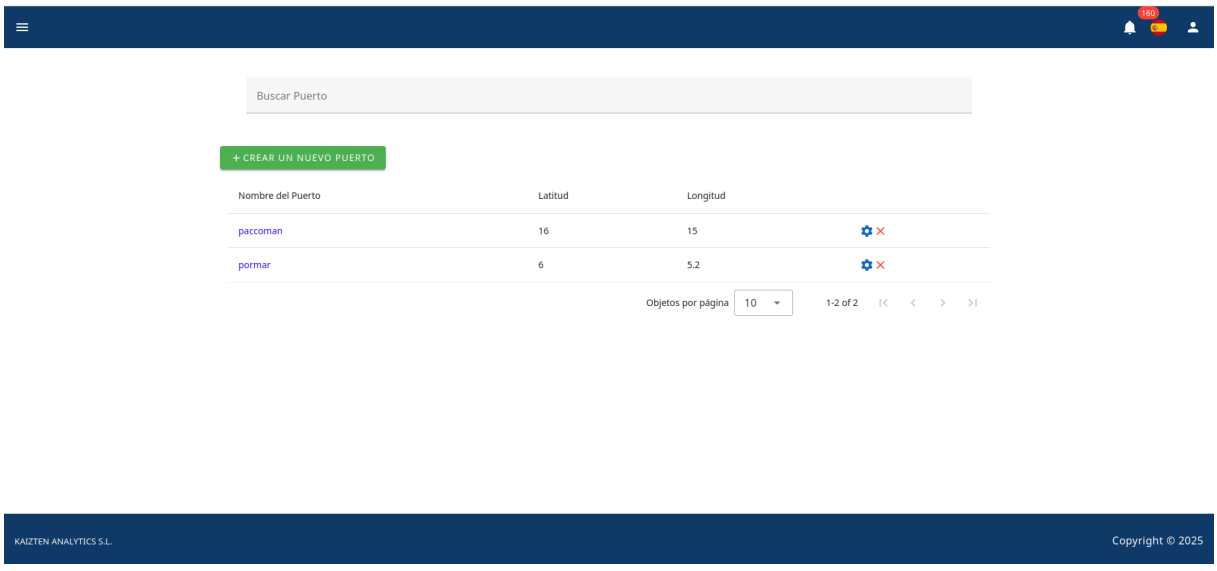

Figura C.4: Página de “Listar Puertos”

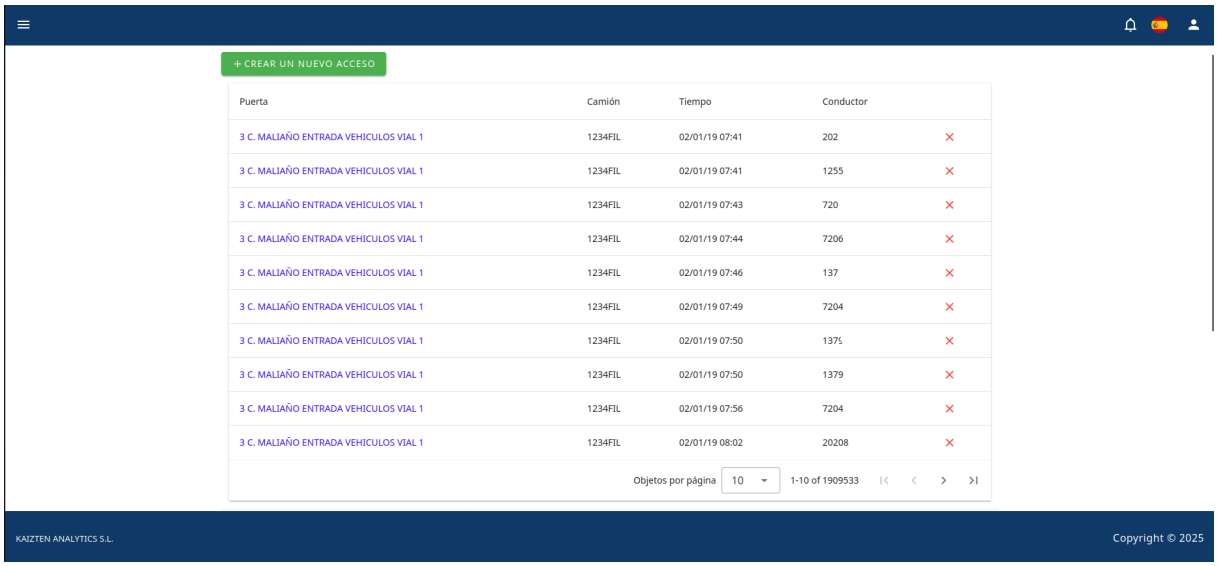

Figura C.5: Página de “Listar Accesos”, registros

Fuera de la tabla, puede estar presente (según el rol del usuario) un botón verde que redirige a la pantalla de creación de una nueva instancia de la entidad. En las vistas de creación, como la mostrada en la figura C.6, es obligatorio rellenar los campos marcados con un asterisco (*). Además, se permite añadir o eliminar atributos opcionales según las necesidades del usuario. También es posible acceder a esta funcionalidad desde el panel de navegación.

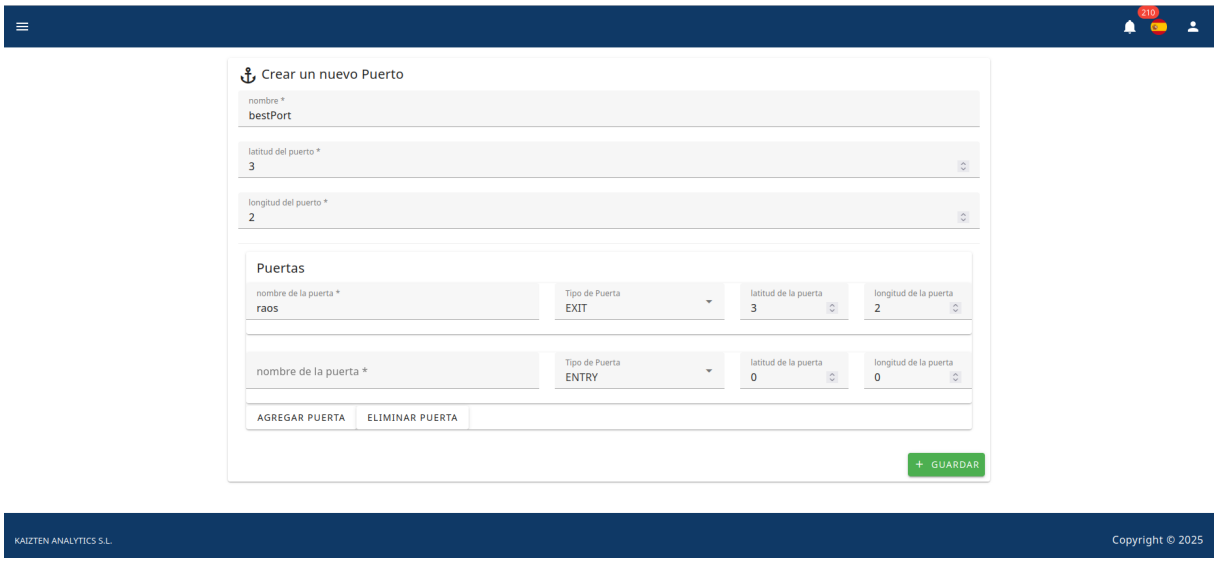

Figura C.6: Página de “Crear Puerto”

Los gráficos de accesos encontrados al final de la vista ver accesos muestran el resumen de datos descrito en el capítulo 4. Esta funcionalidad tiene varias opciones para filtrar los datos que se desean observar por parte de los gestores portuarios o administradores, los filtros son de izquierda a derecha como se muestra en la figura C.7. Puerta escogida, día de la semana escogido y tipo de medición utilizada.

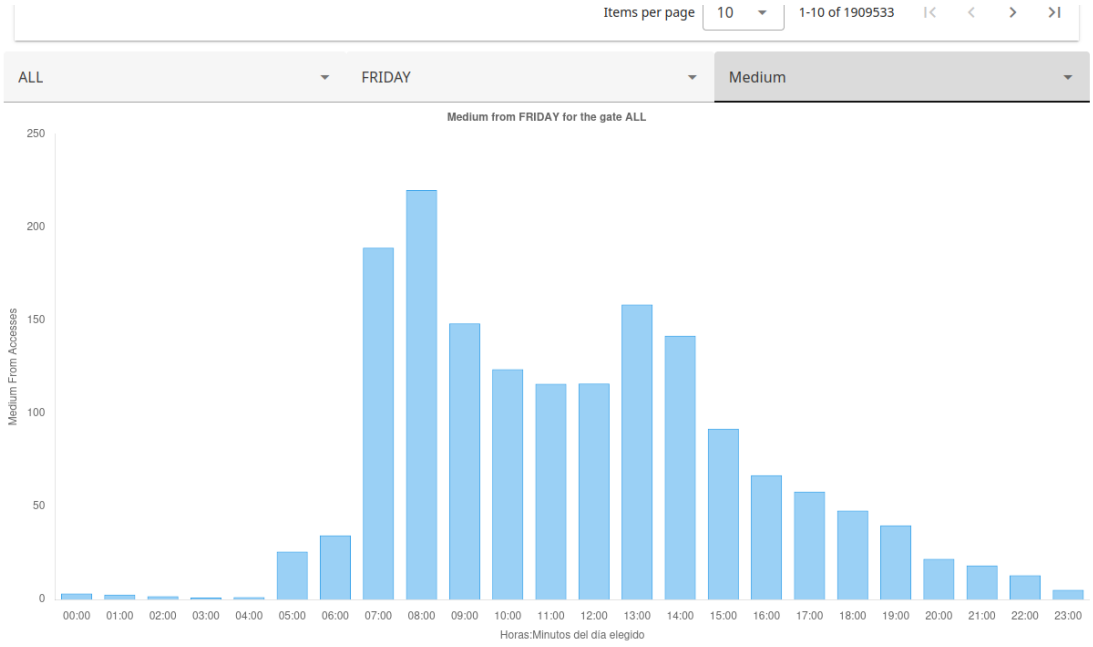

Figura C.7: Página de “Listar Accesos”, resumen

Para crear accesos de forma masiva, se puede utilizar la vista de importación de datos, disponible exclusivamente para administradores. En esta página, el administrador debe seleccionar el puerto que se utilizará durante la operación, así como el formato de fecha y hora del archivo (en caso de que no siga un formato estándar). Luego, puede subir un archivo CSV que contenga las columnas requeridas: Fecha, Acceso, DNI, Nº de Tarjeta y Empresa.

Además, se ofrece la opción de eliminar los accesos existentes antes de importar los nuevos datos. En la figura C.5, se puede observar la gran cantidad de registros presentes en la base de datos, los cuales fueron importados mediante este método.

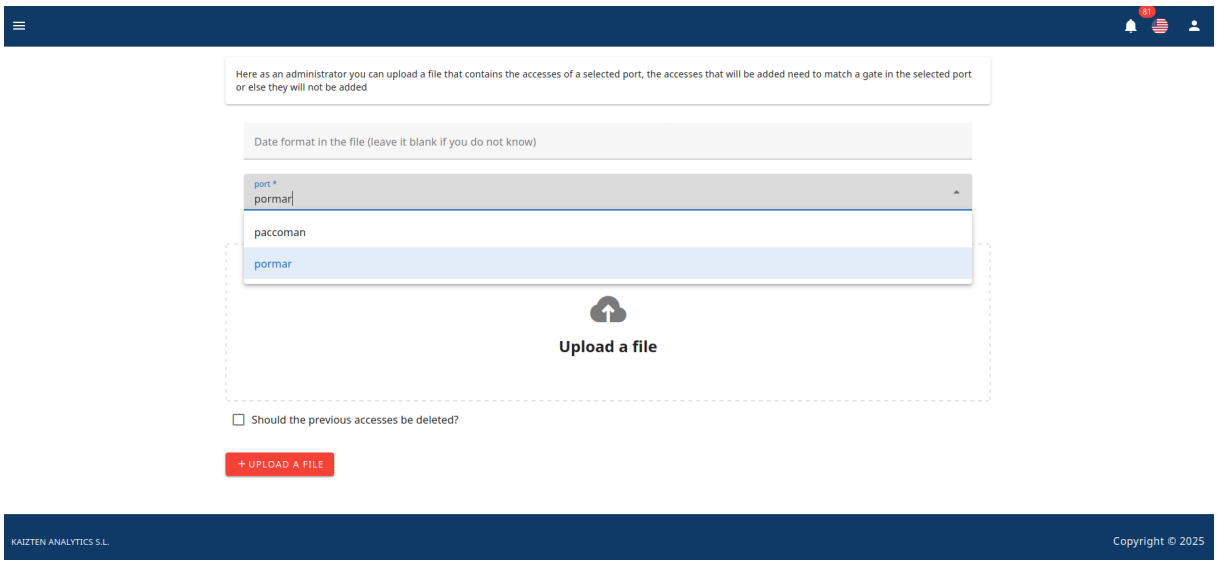

Figura C.8: Página de “Subir Archivo”

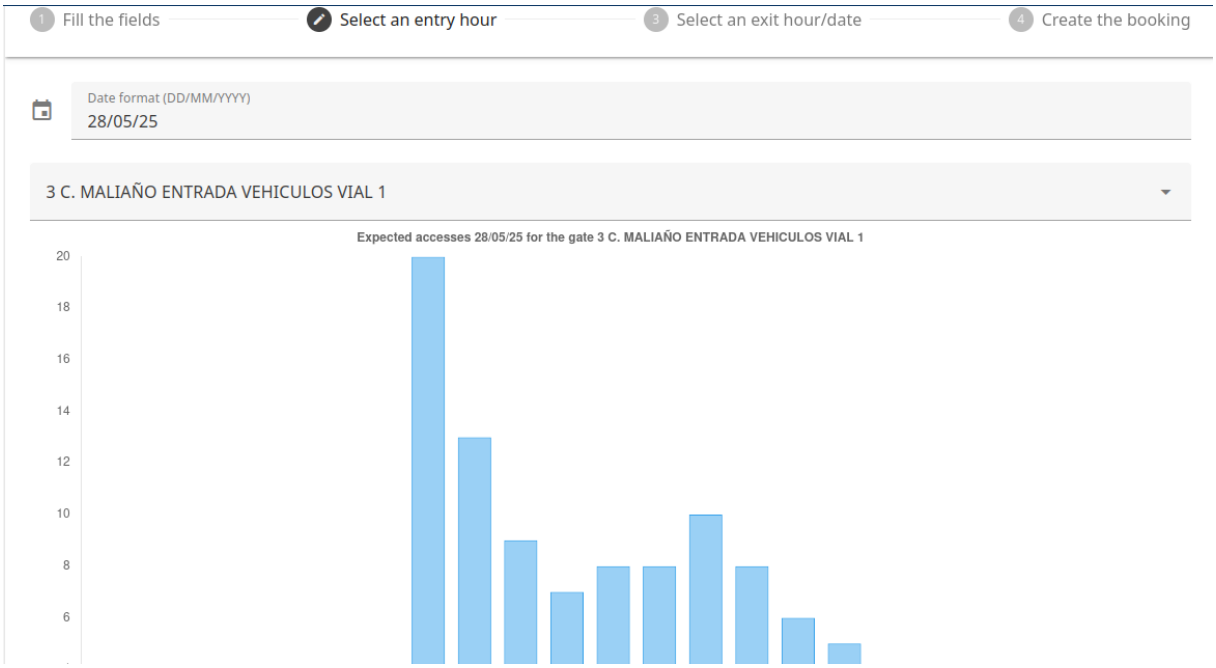

Figura C.9: Página de "Crear Reserva", paso dos

C.4. Creación de reservas

La creación de reservas, a diferencia de otras vistas de creación, se compone de 3 o 4 pasos. La versión antigua, de tres pasos (*old*), utiliza un modelo entrenado para estimar el tiempo de estancia. En cambio, la versión más elaborada, de cuatro pasos, emplea un modelo predictivo de tráfico.

En el primer paso, es necesario completar todos los campos requeridos. A continuación, en el segundo paso, se mostrarán al usuario gráficas como la que aparece en la figura C.9. Según la versión del proceso utilizada, estas gráficas incluirán filtros específicos que permiten visualizar los resultados por según puerta.

El usuario puede seleccionar directamente una barra en la gráfica mostrada, lo que determinará la hora correspondiente a dicha barra. Finalmente, en el último paso, se presenta una vista previa del objeto completo que se va a crear (como se muestra en la figura C.10), permitiendo al usuario realizar los cambios que desee antes de guardar la reserva recién creada.

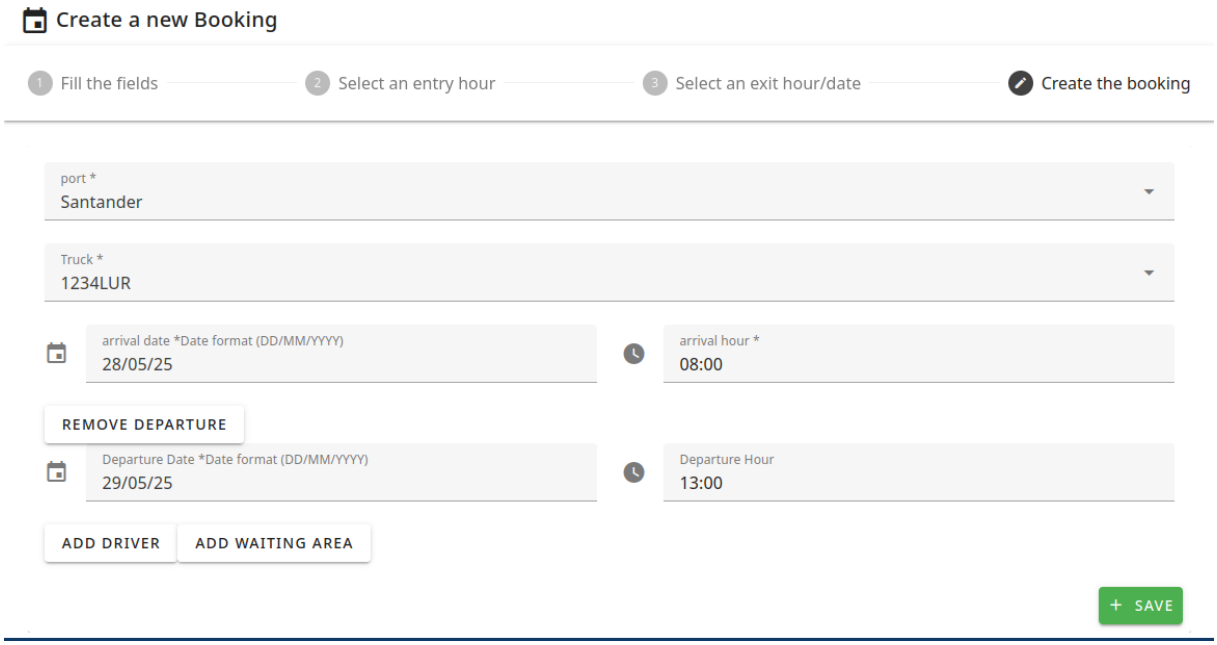

Figura C.10: Página de “Crear Reserva”, paso cuatro

Glosario

back-end Parte del desarrollo web que se encarga de la lógica del servidor, bases de datos y la comunicación entre el servidor y el cliente.. 15, 17, 18, 22, 31, 35

dataset Conjunto de datos utilizado durante las distintas fases desde el analisis hasta su división en el training set y test set. . 39

front-end Parte del desarrollo web que se encarga de la interfaz de usuario y la interacción con el usuario.. 14, 16, 18, 22, 31, 75

full-stack Arquitectura de una aplicación web que abarca tanto el *front-end* (interfaz de usuario) como el *back-end* (servidor, bases de datos y lógica de negocio).. 2, 3, 6, 14–16, 21

machine learning Técnica de la inteligencia artificial que permite a las máquinas aprender de los datos sin ser explícitamente programadas.. 2, 15, 63, 73

Kaggle Kaggle es una plataforma de ML y ciencia de datos, donde los usuarios aprenden a utilizar técnicas de ML y participan en competiciones para resolver problemas reales.. 54

Acrónimos

CRISP-DM Cross Industry Standard Process for Data Mining. 6, 13–15

CRUD Crear, Leer, Actualizar, Eliminar (*Create, Read, Update, Delete*). 23, 24, 36

DNI Documento Nacional de Identidad. 16

ETL *Extract, Transform, Load*. 11

JIT *Just In Time*. 9

ML *Machine Learning*. 2–4, 7, 8, 10–13, 15, 26, 32, 37, 39, 53

ODS Objetivos de Desarrollo Sostenible. 73

SVM Support Vector Machine. 8, 11

TAS *Truck Appointment System*. 9

Bibliografía

[1] A. Paraskevas, M. Madas, V. Zeimpekis y K. Fouskas, «Smart Ports in Industry 4.0: A Systematic Literature Review,» *Logistics*, vol. 8, n.º 1, pág. 28, 2024. DOI: 10.3390/logistics8010028.

[2] B. Behdani, «Port 4.0: a conceptual model for smart port digitalization,» *Transportation Research Procedia*, vol. 74, págs. 346-353, 2023. DOI: 10.1016/j.trpro.2023.11.154.

[3] B. Belmoukari, J.-F. Audy y P. Forget, «Smart port: a systematic literature review,» *European Transport Research Review*, vol. 15, n.º 4, págs. 1-16, 2023. DOI: 10.1186/s12544-023-00581-6.

[4] M. Akbari y T. N. A. Do, «A systematic review of machine learning in logistics and supply chain management: current trends and future directions,» *Benchmarking: An International Journal*, vol. 28, n.º 10, págs. 2977-3005, 2021.

[5] I. Kourounioti, A. Polydoropoulou y C. Tsiklidis, «Development of models predicting dwell time of import containers in port container terminals–an Artificial Neural Networks application,» *Transportation Research Procedia*, vol. 14, págs. 243-252, 2016.

[6] Y. Xie y N. Huynh, «Kernel-based machine learning models for predicting daily truck volume at seaport terminals,» *Journal of transportation engineering*, vol. 136, n.º 12, págs. 1145-1152, 2010.

[7] E. J. C. Suárez, «Tutorial sobre máquinas de vectores soporte (sVM),» *Tutorial sobre Máquinas de Vectores Soporte (SVM)*, vol. 1, págs. 1-12, 2014.

[8] A. Nadi, S. Sharma, M. Snelder, T. Bakri, H. van Lint y L. Tavasszy, «Short-term prediction of outbound truck traffic from the exchange of information in logistics hubs: A case study for the port of Rotterdam,» *Transportation Research Part C: Emerging Technologies*, vol. 127, pág. 103 111, 2021.

[9] Hamburger Hafen und Logistik AG (HHLA), *How machines learn to learn*, Consultado el 6 de junio de 2025, 2020. dirección: https://hhla.de/.

[10] MC Valnera, *Puertos 4.0: Innovación en el sector portuario*, Disponible en: https://mcvalnera.com/, 2023.

[11] N. Huynh, D. Smith y F. Harder, «Truck appointment systems: where we are and where to go from here,» *Transportation Research Record*, vol. 2548, n.º 1, págs. 1-9, 2016.

[12] M. D. Gracia, J. Mar-Ortiz y M. Vargas, «Truck Appointment Scheduling: A Review of Models and Algorithms,» *Mathematics*, vol. 13, n.º 3, pág. 503, 2025.

[13] A. Abdelmagid, M. Gheith y A. Eltawil, «A comprehensive review of the truck appointment scheduling models and directions for future research,» *Transportation Research Part E: Logistics and Transportation Review*, vol. 146, pág. 102 126, 2021. DOI: 10.1016/j.tre.2021.102126. dirección: https://doi.org/10.1016/j.tre.2021.102126.

[14] D. Konur y M. M. Golias, «Cost-stable truck scheduling at a cross-dock facility with unknown truck arrivals: A meta-heuristic approach,» *Transportation Research Part E: Logistics and Transportation Review*, vol. 49, n.º 1, págs. 71-91, 2013.

[15] H. Lee, I. Chatterjee y G. Cho, «AI-powered intelligent seaport mobility: enhancing container drayage efficiency through computer vision and deep learning,» *Applied Sciences*, vol. 13, n.º 22, pág. 12 214, 2023.

[16] O. Doerr, «Indicadores de rendimiento (KPI) para productividad y eficiencia en puertos de Latinoamérica y el Caribe,» Comisión Económica para América Latina y el Caribe (CEPAL), inf. téc., 2020. dirección: https://www.cepal.org/.

[17] H. Chen, «Predicting Truck Turn Times at the Port of Rotterdam using LSTM Networks,» Tesis de mtría., TU Delft, 2025.

[18] M. Jahangard, Y. Xie e Y. Feng, «Leveraging machine learning and optimization models for enhanced seaport efficiency,» *Maritime Economics & Logistics*, págs. 1-42, 2025.

[19] Bilbaoport, *El Puerto de Bilbao acelera su automatización de procesos con AllRead*, 2025. dirección: https://www.bilbaoport.eus.

[20] A. Gutiérrez, «Simu-Port: Optimización de operaciones logístico-portuarias mediante inteligencia artificial y simulación,» *Revista Mar*, 2025, Accedido: 2025-05-14. dirección: https://revistamar.seg-social.es/-/iapuertos655.

[21] Ingeniería y Soluciones Informáticas del Sur S.L. (ISOIN), *Optimización de operaciones logístico-portuarias para tráficos RO-PAX mediante la hibridación de simulación e inteligencia artificial*, 2025. dirección: https://www.isoin.es/.

[22] C. Shearer, «The CRISP-DM model: the new blueprint for data mining,» *Journal of data warehousing*, vol. 5, n.º 4, págs. 13-22, 2000.

[23] C. Larman y V. R. Basili, «Iterative and incremental developments. a brief history,» *Computer*, vol. 36, n.º 6, págs. 47-56, 2003.

[24] A. Cockburn, «Hexagonal architecture,» *The Pattern: Ports and Adapters*, 2005.

[25] M. Potel, «MVP: Model-View-Presenter the Taligent programming model for C++ and Java,» *Taligent Inc*, vol. 20, 1996.

[26] M. Fuksa, S. Speth y S. Becker, «MVVM Revisited: Exploring Design Variants of the Model-View-ViewModel Pattern,» en *International Conference on Enterprise Design, Operations, and Computing*, Springer, 2024, págs. 163-181.

[27] E. Evans, *Domain-driven design: tackling complexity in the heart of software*. Addison-Wesley Professional, 2004.

[28] L. Breiman, «Random forests,» *Machine learning*, vol. 45, págs. 5-32, 2001.

[29] T. Chen, T. He, M. Benesty et al., «Xgboost: extreme gradient boosting,» *R package version 0.4-2*, vol. 1, n.º 4, págs. 1-4, 2015.

ULPGC
Universidad de
Las Palmas de
Gran Canaria

ESCUELA DE INGENIERÍA
INFORMÁTICA