Universitat Oberta de Catalunya (UOC)

Máster Universitario en Ciencia de Datos (*Data Science*)

TRABAJO FINAL DE MÁSTER

Área: 4

Estudio y evaluación de algoritmos de procesamiento de datos inalámbricos para la estimación de la localización y ocupación en interiores.

Study and evaluation of wireless data processing algorithms for indoor location and occupancy estimation.

Autor: Ginés Molina Abril

Tutor: Joaquín Torres Sospedra

Profesor: Antonio Lozano Bagén

Barcelona, 5 de junio de 2022

Créditos/Copyright

Ginés Molina Abril es el autor de esta obra que ha sido desarrollada como proyecto final del Máster Universitario de Ciencia de Datos de la Universitad Oberta de Catalunya. Ha contado con la dirección del profesor **Dr. Joaquín Torres-Sospedra** y la colaboración del profesor **Dr. Alejandro S. Martinez-Sala**.

Esta obra está bajo una licencia de Creative Commons Reconocimiento - NoComercial - SinObraDerivada 4.0 Internacional.

El repositorio digital con el código desarrollado en este proyecto está protegido por copyright con todos los derechos de autor reservados, sin concesión de licencia de uso, por lo que no se permite su difusión, manipulación o uso de ningún tipo sin la autorización expresa del autor.

Copyright (c) Ginés Molina Abril - Todos los derechos reservados 2022

FICHA DEL TRABAJO FINAL

Título del trabajo:	**Estudio y evaluación de algoritmos de**
	procesamiento de datos inalámbricos para la
	estimación de la localización y ocupación en
	interiores.
Nombre del autor:	Ginés Molina Abril
Nombre del colaborador/a docente:	Joaquín Torres Sospedra
Nombre del colaborador/a externo:	Alejandro Martínez Sala
Nombre del PRA:	Antonio Lozano Bagén
Fecha de entrega (mm/aaaa):	06/2022
Titulación o programa:	Máster Universitario en Ciencia de Datos
Área del Trabajo Final:	Área 4
Idioma del trabajo:	Español
Palabras clave	Indoor Positioning Systems,
	Wi-Fi fingerprint,
	MLOps

Agradecimientos

Lo primero y más importante, me gustaría poder agradecer a mi pareja **Diana** por su paciencia y por el apoyo que me ha mostrado en el desarrollo del Máster Universitario desde el primer día. Ella sabe más que nadie los esfuerzos que hemos tenido que hacer para lograr llegar al final del trayecto de la manera en la que se ha llegado, aprovechando al máximo la oportunidad de aprendizaje en un área que me apasiona y logrando los mejores resultados. No ha sido un camino fácil y ahora que ha pasado el tiempo, no hay mayor orgullo que sentir que el trabajo y el esfuerzo han dado sus frutos. Han sido tiempos muy difíciles que había que gestionar de la mejor manera y que nos ha hecho crecer como personas, como pareja, como equipo. Siempre te estaré agradecido por lo que haces por mí y por nuestra familia.

Me gustaría también agradecer todo el apoyo que me han mostrado siempre mis padres **Ginés** y **Fina**, y mi hermana **Joanna**. Ellos están siempre en todas las aventuras que me planteo y son los primeros en ofrecerme todo lo que pueden para ayudarme a lograr mis metas. Me han enseñado a salir siempre reforzado de los momentos más complicados y sin duda que les debo todo lo que soy y todo lo que he conseguido.

Aprovecho también para agradecer a toda **la gente que confía en mí** y me ha ofrecido su ayuda de manera desinteresada. Me han enseñado a creer en mí y a evolucionar como persona y como profesional, buscando siempre la mejor parte de mí.

Por último, agradecer a **Alejandro** y a **Joaquín** por acompañarme en este proyecto que dará por finalizado el máster y que me ofrece la oportunidad de continuar con el aprendizaje en este sector planteándome posibles retos de futuro que ilusionan.

Resumen

Si hablamos de posicionamiento en exteriores podríamos afirmar que GPS es la tecnología estándar. Pero el ser humano es una especie que pasa la mayor parte del tiempo en espacios interiores, y ahí GPS cuenta con algunas limitaciones. Para solucionarlo nacen los sistemas de posicionamiento en interiores o IPS. Estos sistemas pueden aprovechar infraestructuras ya creadas de Wi-Fi o Bluetooth Low Energy (BLE) para dar servicio a muy bajo coste, con un bajo consumo de energía, con mayor flexibilidad y mayor compatibilidad de dispositivos. Estos sistemas además pueden funcionar como sistemas IoT heterogéneos usando ambas técnicas y obteniendo grandes resultados, potenciando sus ventajas para mitigar los problemas asociados a cualquier sistema inalámbrico de interiores como pueden ser las interferencias.

Partiendo de los trabajos anteriores [42] [25] [15], se propone el estudio y la evaluación de familias de algoritmos de estimación de la localización (posición y nivel de área) aplicados a diversos escenarios utilizando las técnicas descritas anteriormente. El objetivo final es el desarrollo de una herramienta "end-to-end" llevada a producción que sea capaz de predecir la localización de un dispositivo a través de su huella digital o "fingerprint". Tal y como pasa en el ciclo de vida de un producto software y con un mayor número de empresas abriéndose camino en la aplicación del aprendizaje automático en sus productos y servicios, aparece el concepto de deuda técnica. La deuda técnica es el efecto que puede tener una decisión en las etapas de diseño de cara a generar un mayor esfuerzo de mantenimiento o trabajo en el futuro.

Por ello, y siguiendo el enfoque MLOps, se deberá tener presente en todo momento las decisiones de diseño que estén orientadas a ofrecer la posibilidad de escalar en número de componentes del sistema, ofrecer una mayor versatilidad y ofrecer herramientas de monitorización que permitan seguir la evolución de las distintas arquitecturas de modelos y lo que se conoce como Entrenamiento Contínuo (CT). A este concepto se le une los ya conocidos como Integración Contínua (CI) y Entrega Contínua (CD), ya presentes en cualquier ciclo de vida de productos software. El sistema propuesto deberá ser fácilmente integrado en análisis de flujos de datos constantes a través de sistemas IoT y podrá evolucionar a arquitecturas más complejas como arquitecturas Transfer Learning o Incremental Learning.

Palabras clave: Indoor Positioning Systems (IPS), Wi-Fi Fingerprinting, MLOps, K-means Clustering, K-nearest neighbors (k-NN), Support vector machine (SVM), Multinomial logistic regression, Continuous Training (CT), ETL Pipeline, IoT infrastructure, Concept drift, MLFlow

Índice general

Resumen vii

Índice ix

Listado de Figuras xiii

Listado de Tablas xv

1. Introducción 1

1.1. Descripción 1

1.2. Motivación, interés y relevancia 3

1.3. Definición de objetivos 3

1.4. Planificación, workflow y metodología 4

2. Estado del arte 9

2.1. Contexto e historia 9

2.2. Tecnologías y Sistema de Posicionamiento en Interiores 11

2.2.1. Técnicas para cálculo de la posición 11

2.2.2. Tecnologías 16

2.3. Aplicaciones de posicionamiento en interiores 18

2.3.1. Soluciones de proximidad 18

2.3.2. Soluciones de posicionamiento 19

2.4. Localización en interiores mediante Wi-Fi RSS Fingerprinting 21

2.5. Localización en interiores mediante BLE 21

2.6. Modelado 22

2.6.1. Algoritmos de clustering 22

2.6.2. Algoritmo KNN 23

2.6.3. Regresión logística multinomial 24
2.6.4. Algoritmo SVM 25
2.6.5. Otros algoritmos y arquitecturas 25

3. Diseño, modelado y desarrollo del proyecto 27
3.1. Entorno y descripción del dataset 27
3.2. Deuda técnica en proyectos ML 28
3.2.1. Enfoque MLOps 29
3.3. Tecnologías y sistemas 30
3.4. Flujo de datos 33
3.5. Cross-Validation y métricas de éxito 39

4. Resultados y predicciones 41
4.1. Predicciones - Edificio 41
4.2. Predicciones - Planta 44
4.3. Arquitectura de modelos - Clustering 48
4.4. Predicciones - Localización 49
4.5. Infraestructura y predicciones 52

5. Trabajo futuro, debate y conclusiones 55
5.1. Adaptación a dataset Bluetooth Low Energy 55
5.2. Transfer Learning e Incremental Learning - IoT 56
5.3. Monitorización y alertas 56
5.4. Conclusiones 57

A. Infograma de contextualización 59

B. Cronograma de tareas 61

C. Anaconda, Python e instalación de librerías asociadas 71

D. Instalación y uso de Ploomber & MLFlow 73

E. Informe y registro de eventos 75

F. Diagrama y flujo de datos 79

G. Informe sobre el dataset 81

H. Plataforma MLFlow 87

Bibliografía **89**

Índice de figuras

1.1. Planificación del proyecto 5
1.2. Metodología de trabajo 7
1.3. Planificación final del proyecto 8

2.1. Estudio de la NHAPS muestra que los estadounidenses pasan el 86,9 % del tiempo en espacios interiores, mas otro 5,5 % dentro de un vehículo. 10
2.2. Lugares donde pasan el tiempo encuestados por la NHAPS a lo largo de 24 horas. 10
2.3. Tabla resumen con las características importantes en los sistemas IPS y sus tecnologías asociadas. 12
2.4. Ejemplo de flow-chart del método fingerprinting usando Wi-Fi. 13
2.5. Ejemplo del proceso de estimación de la localización con Wi-Fi fingerprinting. . 14
2.6. Triangulación de potencias para, por ejemplo, un sistema RTLS. 14
2.7. Técnica de posicionamiento según el ángulo del receptor. 15
2.8. Ejemplo de AoD utilizando Bluetooth. 15
2.9. Algunas de las aplicaciones de las tecnologías y métodos de localización. 18

3.1. Deuda técnica en sistemas de ML - Sólo una pequeña parte de los sistemas de ML del mundo real está compuesta por el código de ML. La infraestructura necesaria es amplia y compleja. 28
3.2. Comparativa de tareas entre un DevOps y MLOps 30
3.3. Diagrama global de tecnologías 32
3.4. Scatter Matrix de los atributos 34
3.5. Correlaciones - WAPs 34
3.6. KMeans Clustering in Building 0 36
3.7. Ploomber plot - Autogenerado 36

4.1. Predicciones de edificio 43
4.2. Predicciones de Planta - Edificio 0 45
4.3. Predicciones de Planta - Edificio 1 46

4.4. Predicciones de Planta - Edificio 2 48
4.5. Pantalla principal de experimentos - MLFlow 53

A.1. Diagrama y contextualización de las diferentes etapas de la propuesta. 60

B.1. Workflow - Sección 1 62
B.2. Workflow - Sección 2 63
B.3. Workflow - Sección 3 64
B.4. Workflow - Sección 4 65
B.5. Workflow - Sección 5 66
B.6. Workflow - Sección 6 67
B.7. Workflow - Sección 7 68

F.1. Diagrama global de funcionamiento del sistema 80

G.1. Overview Pandas Profiling Report 81
G.2. Alerts Pandas Profiling Report 82
G.3. Reproduction Pandas Profiling Report 82
G.4. Variables Pandas Profiling Report 83
G.5. Interactions Pandas Profiling Report 83
G.6. Correlations Pandas Profiling Report 84
G.7. Missing Values Pandas Profiling Report 84
G.8. Sample Pandas Profiling Report 85
G.9. Duplicate Rows Pandas Profiling Report 85

H.1. Pantalla principal de modelos - MLFlow 87
H.2. Vista del modelo en específico - MLFlow 88
H.3. Vista de la Versión del modelo - MLFlow 88
H.4. Detalles de la Versión del modelo - MLFlow 89
H.5. Detalles de la versión del modelo - Predicciones 89

Índice de cuadros

4.1. Predicciones de Edificio 42
4.2. Predicciones de Planta - Edificio 0 44
4.3. Predicciones de Planta - Edificio 1 46
4.4. Predicciones de Planta - Edificio 2 47
4.5. Predicciones de Localización - KNN - Edificio 0 50
4.6. Predicciones de Localización - KNN - Edificio 1 50
4.7. Predicciones de Localización - KNN - Edificio 2 50
4.8. Predicciones de Localización - LR - Edificio 0 51
4.9. Predicciones de Localización - LR - Edificio 1 51
4.10. Predicciones de Localización - LR - Edificio 2 51
4.11. Predicciones de Localización - SVM - Edificio 0 52
4.12. Predicciones de Localización - SVM - Edificio 1 52
4.13. Predicciones de Localización - SVM - Edificio 2 52

Capítulo 1

Introducción

1.1. Descripción

La capacidad de poder ubicar objetos o personas en un mundo conectado se ha convertido en una excelente oportunidad de negocio. Existe información que puede aprovecharse para brindar un mejor servicio o diseñar sistemas que puedan ofrecer al usuario valor extra y una mejor experiencia de usuario mucho más adaptada a sus necesidades. Uno de los factores importantes a tener en cuenta es el auge de las infraestructuras IoT que hacen posible registrar un mayor volumen de información. Toda esta información registrada por estos sistemas puede llegar a ofrecer un valor adicional sobre el comportamiento de los usuarios en base a su ubicación.

Hasta ahora, la ubicación en espacios abiertos podía quedar registrada a través de señales GNSS, como GPS. Sin embargo, este sistema GPS presenta conflictos por sus limitaciones en espacios interiores, por ejemplo, debido a la cobertura o a la identificación de los distintos pisos o niveles de altura en los edificios. Por tanto, el interés de este proyecto se basa en el análisis de la problemática asociada al desarrollo de este tipo de arquitecturas como solución de posicionamiento y seguimiento en interiores. Esta solución será llevada a cabo desde el análisis teórico de la señal hasta la implementación del sistema en el mundo real o llevada a producción. El análisis se llevará a cabo en un escenario en el que se cuenta con registros de la huella digital Wi-Fi de los usuarios pero sin perder de vista la aplicación final en un sistema heterogéneo IoT en el que se pueda utilizar también información registrada a través de Bluetooth Low Energy (BLE) u otras tecnologías de posicionamiento.

Partiendo de estas técnicas y este escenario, se propone un análisis de resultados de las distintas arquitecturas desarrolladas y un modelo o sistema llevado a producción que esté basado en la nube. En este sistema se podrán solicitar las predicciones según los niveles de potencia

registrados por los puntos de acceso desde cualquier dispositivo móvil. Este análisis de resultados será alimentado de forma iterativa para poder determinar qué modelos han funcionado mejor según qué tipo de predicción, en qué espacio y según qué características del espacio. Además, en el proyecto se desarrollará un proceso de extracción, transformación y carga de datos (ETL) afín a lo que un Ingeniero de datos desarrollaría para preparar los datos pensando en un consumo final a través de los modelos teniendo en cuenta las distintas características de los valores de potencia y la naturaleza de los mismos. Esta etapa es importante para poder acercar la propuesta a un escenario real en el que se deban tener en cuenta limitaciones o restricciones. Este arquitectura propuesta debería poder ser capaz de analizar escenarios con datos fijos de entrenamiento a través de ficheros locales, un flujo de datos constante y una fuente de datos en la nube para cargar por batches o lotes. En esta etapa se hará especial énfasis en cambios de infraestructura para comprobar cómo podría afectar esto en los resultados de los modelos y cómo se podría paliar.

Por último, se desarrollarán distintos modelos y arquitecturas con el objetivo de clasificar y predecir la localización (edificio, planta y habitación o zona). Se propone la realización del proceso holístico e iterativo de validación "a ciegas" con el objetivo de poder adquirir el conocimiento necesario para poder predecir sin saber realmente las predicciones. Se contará por tanto con un dataset de entrenamiento, otro de validación y otro dataset final de evaluación con el objetivo de ser evaluado por un agente externo al sistema. Esto será muy importante para evitar que el análisis quede más orientado a intentar ajustarse a un escenario en concreto, en vez de encontrar las particularidades, problemas o limitaciones que pueden surgir en cualquiera de ellos. Finalmente, estos modelos y arquitecturas serán comparados y se deberá dar respuesta a las preguntas siguientes:

- ¿Cualés son las ventajas y desventajas de la utilización de sistemas BLE y Wi-Fi en sistemas de posicionamiento en interiores?

- ¿Qué pasa si la infraestructura cambia? ¿Se cuenta con una arquitectura lo suficientemente flexible ante esos cambios? ¿Qué se podría hacer para proporcionar un sistema fiable a estos cambios en una arquitectura ya puesta en producción?

- ¿Cómo se pueden adaptar los algoritmos a los posibles cambios? ¿Hay alguna manera de poder realizar ajustes y que de manera sencilla se puedan pasar a producción los cambios sin apenas esfuerzo?

- ¿Cómo se podría ajustar el proceso al trabajar con una infraestructura IoT real?

- ¿Qué respuesta ofrecen estos sistemas y estos algoritmos funcionando en diferentes escenarios o según cada tipo de predicción?
- ¿Se podrían obtener datos en tiempo real sobre las predicciones? ¿Se podría aplicar concept drift? ¿Cómo se podría ajustar esta arquitectura para recibir un flujo de datos continuo?
- ¿Cómo afectan a los resultados los hiperparámetros de los algoritmos según tipo de información?

1.2. Motivación, interés y relevancia

Siendo Ingeniero Técnico de Telecomunicaciones con experiencia como Ingeniero de Software y desarrollador de soluciones cloud, me ha parecido interesante cerrar el círculo de mi aprendizaje poniendo cada ingrediente de mi experiencia académica y profesional en este proyecto. En primer lugar, siempre he sentido interés en el modo en el que las máquinas se comunican y siempre me ha parecido interesante saber, de manera física, cómo se propaga la señal y cómo actúan los protocolos de comunicación. Internet hizo conectar el mundo y por ello, decidí enfocar mi carrera profesional en el desarrollo web ampliando mi foco de soluciones hardware a soluciones software. Esto me ofrece un campo de visión amplio "end-to-end" en el que puedo ser capaz de diseñar y desarrollar un servicio o producto tecnológico desde el más bajo nivel físico (hardware) o de infraestructura, hasta la presentación final (software).

Entendiendo lo que supone un servicio o producto tecnológico y con la revolución que estamos viviendo en el análisis de los datos, decidí ampliar mi conocimiento en esta dirección. De esta manera, puedo entender la tecnología desde distintos puntos de vista y profundizar en desarrollos que requieran conocimiento experto en varias de estas etapas. Por ello, se ha propuesto un proyecto que complementa mi conocimiento en cada una de estas etapas para así, poder enfocarlo en lo que me gustaría que fuera mi línea de investigación futura a través de un doctorado.

1.3. Definición de objetivos

1. Entender las particularidades, ventajas e inconvenientes del uso de técnicas BLE y Wi-Fi fingerprinting en sistemas IPS.
2. Comprender cómo poder utilizar ambas técnicas en un escenario IoT híbrido y cambiante que requiera de una infraestructura y una puesta en producción flexible.

3. Desarrollo de una herramienta que se encargue del proceso de extracción, transformación y carga de los datos en paralelo con una apropiada limpieza de los datos y gestión de valores nulos y atípicos.

4. Entender limitaciones que podrían tener los sistemas IoT reales ante este sistema propuesto y aplicación de técnicas apropiadas en el diseño del sistema para facilitar la adaptaciópn del sistema al uso de Spark Streaming y Structured Streaming.

5. Utilización de los modelos k-NN, KMeans, DBSCAN, SVM y regresión logística multiclase para clasificar las muestras y validarlas "a ciegas" con ejecución y carga secuencial y en paralelo haciendo uso del registro de modelos a través de herramientas MLOps.

6. Comparativa de modelos o arquitecturas de modelos según distintos hiperparámetros y utilización de técnicas como la validación cruzada.

7. Descubrimiento de particularidades según los distintas variables objetivo a predecir y aplicación de modelos sobre los clusters resultantes para refinar los resultados por áreas.

1.4. Planificación, workflow y metodología

La planificación inicial y general del proyecto queda dividida en las fases temporales según muestra la figura 1.1. Cada etapa de la planificación también está representada en un infograma del Anexo A.1, mostrando en líneas generales un diagrama de las etapas principales y la propuesta. Este conjunto de procesos y etapas definen el proyecto a gran escala como si se tratara de una serie de componentes interconectados.

El "Workflow" o flujo de trabajo final se ha establecido según la consecución de objetivos o deadlines en un componente Timeline configurado en la plataforma de Notion.so. Cada objetivo ha sido dividido en tareas y marcadas en la agenda con dos etiquetas que hacen referencia al estado de la tarea y a la entrega asociada. Estas tareas además cuentan con una fecha de inicio y fin para poder tener en cuenta los recursos temporales necesarios para llevarla a cabo y el esfuerzo que requerían para plantear objetivos realistas y hacer una buena gestión del tiempo disponible. Se muestra en detalle el flujo de trabajo en las tareas en el Anexo B.

El objetivo principal es que, de forma iterativa, holística y progresiva se vaya profundizando y reinvirtiendo el conocimiento adquirido sobre el mismo conjunto de procesos. Los procesos o etapas son los siguientes:

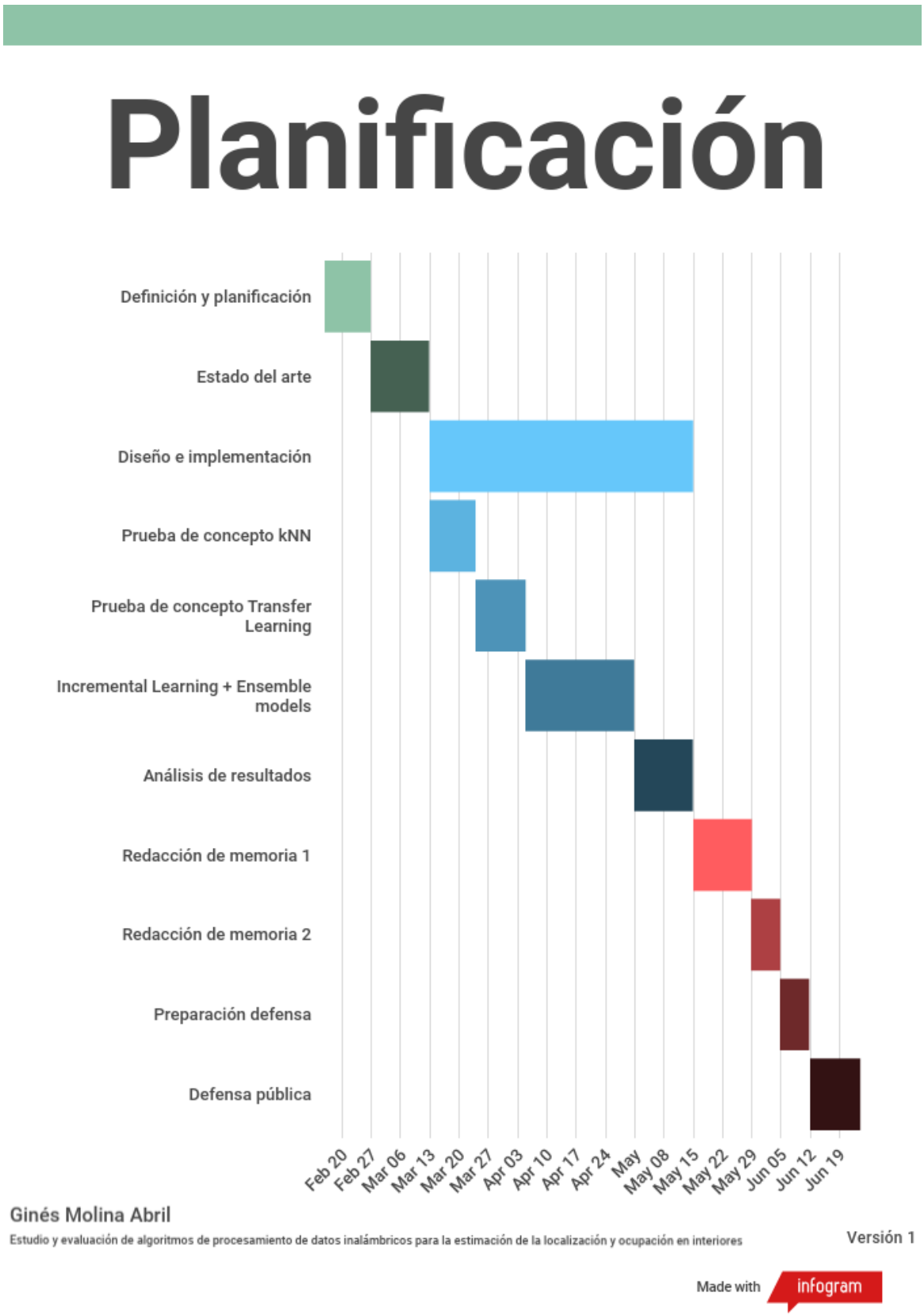


Figura 1.1: Planificación del proyecto

- **Técnicas** - Ventajas e inconvenientes de la utilización de técnicas BLE, Wi-Fi Fingerprint o híbrido.

- **Contexto** - Análisis de características e infraestructuras en Indoor Positioning Systems (IPS).

- **Bases de datos** - Análisis de tipos y modelos de datos, técnicas aplicadas, arquitecturas y objetivos de los distintos escenarios propuestos para ofrecer respuestas apropiadas según la información que se obtiene de la infraestructura.

- **ETL Pipeline** - Análisis de modos de realizar la Extracción, Transformación y Carga de datos (ETL). Se definen labores concretas afines a la Ingeniería de Datos (Data Engineering) en los que se analizan todos los escenarios de carga de datos previos: datos fijos, datos en el cloud o por lotes (batches).

- **Modelos** - Foco principal en la aplicación de distintos modelos y arquitecturas. Comparativa de modelos según los distintos escenarios propuestos: algoritmo k-NN, algoritmos transfer learning, algoritmos incremental learning y técnicas ensemble models (sistema de votación con varias máquinas entrenadas).

- **Resultados** - Ofrecer respuestas a las preguntas propuestas.

Una vez marcados los objetivos al inicio del proyecto en la lista de objetivos y tareas, se establece un enfoque basado en una investigación previa y un estudio de herramientas que facilitan el correcto desarrollo en cada una de las etapas del proyecto. Esta investigación además de establecer el enfoque principal basado en herramientas actuales nos marca el progreso del desarrollo de cada funcionalidad o componente.

Con estos nuevos componentes o funcionalidades realizados, al mismo tiempo, surgen una serie de problemas a resolver que no han sido tenidos en cuenta y se muestran por el mismo desarrollo del proyecto de manera natural. Estos problemas o limitaciones dan lugar a una nueva investigación y un nuevo estudio de las funcionalidades que podrían solventar estos problemas, dando lugar de nuevo un cambio de enfoque.

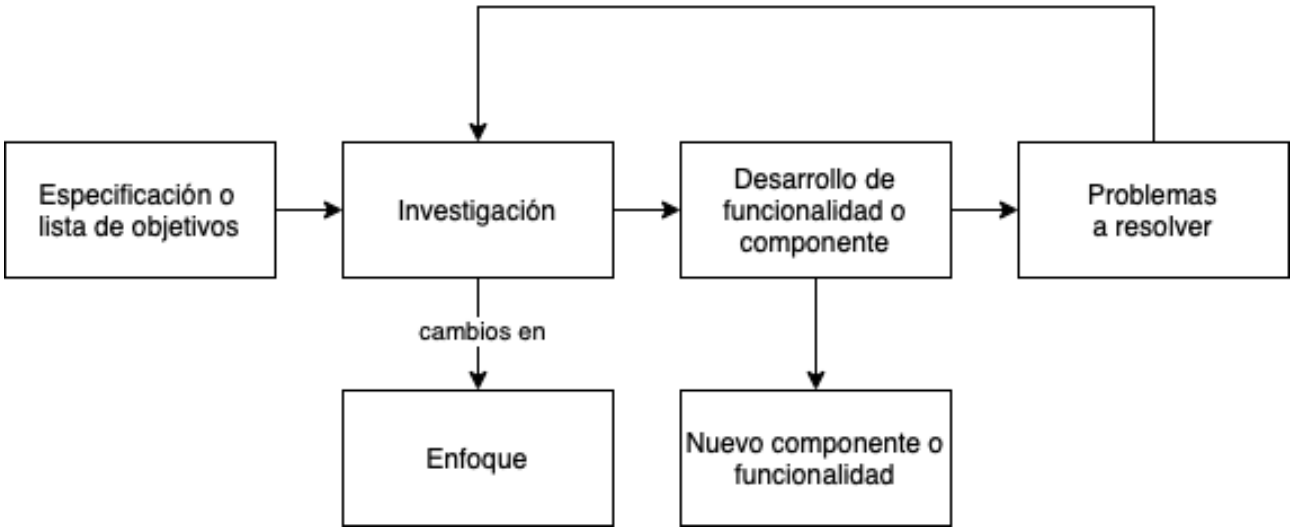


Figura 1.2: Metodología de trabajo

El proceso anterior, tal y como se muestra en la figura 1.2, se repite de manera holística e iterativa dando lugar a un conocimiento profundo de las necesidades, limitaciones y buenas prácticas en este tipo de desarrollos en el mundo real. En la figura 1.3 se muestra la planificación final fruto de los cambios que se han hecho en base a la metodología de trabajo anterior.

En primer lugar, destacar que se ha optado por enfocar el proyecto en una base sólida en cuanto a su arquitectura y su puesta en producción de manera flexible por motivos que posteriormente se describirán en detalle. La planificación inicial, por tanto, sufre algunos cambios en la etapa de diseño e implementación (marcado con el cuadro rojo en la figura 1.3). Además de la asignación de un mayor número de recursos temporales al diseño y desarrollo de la arquitectura, se modifican otras dos etapas más y se opta por asignar más recursos al desarrollo del componente de Clustering y al desarrollo de las arquitecturas de modelos que ofrecerán las distintas predicciones, dejando así la implementación de arquitecturas más complejas para el trabajo futuro, que será fruto de la unión de estos componentes previos ya desarrollados y puestos en producción. Se debe tener en cuenta que estos cambios de enfoque vienen dados desde la puesta en marcha del diseño de la arquitectura que iba a ser propuesta, dejando para un estudio posterior la implementación de arquitecturas Transfer Learning e Incremental Learning, que estarán soportadas por esta arquitectura sobre todo por la flexibilidad que ofrece el sistema propuesto.

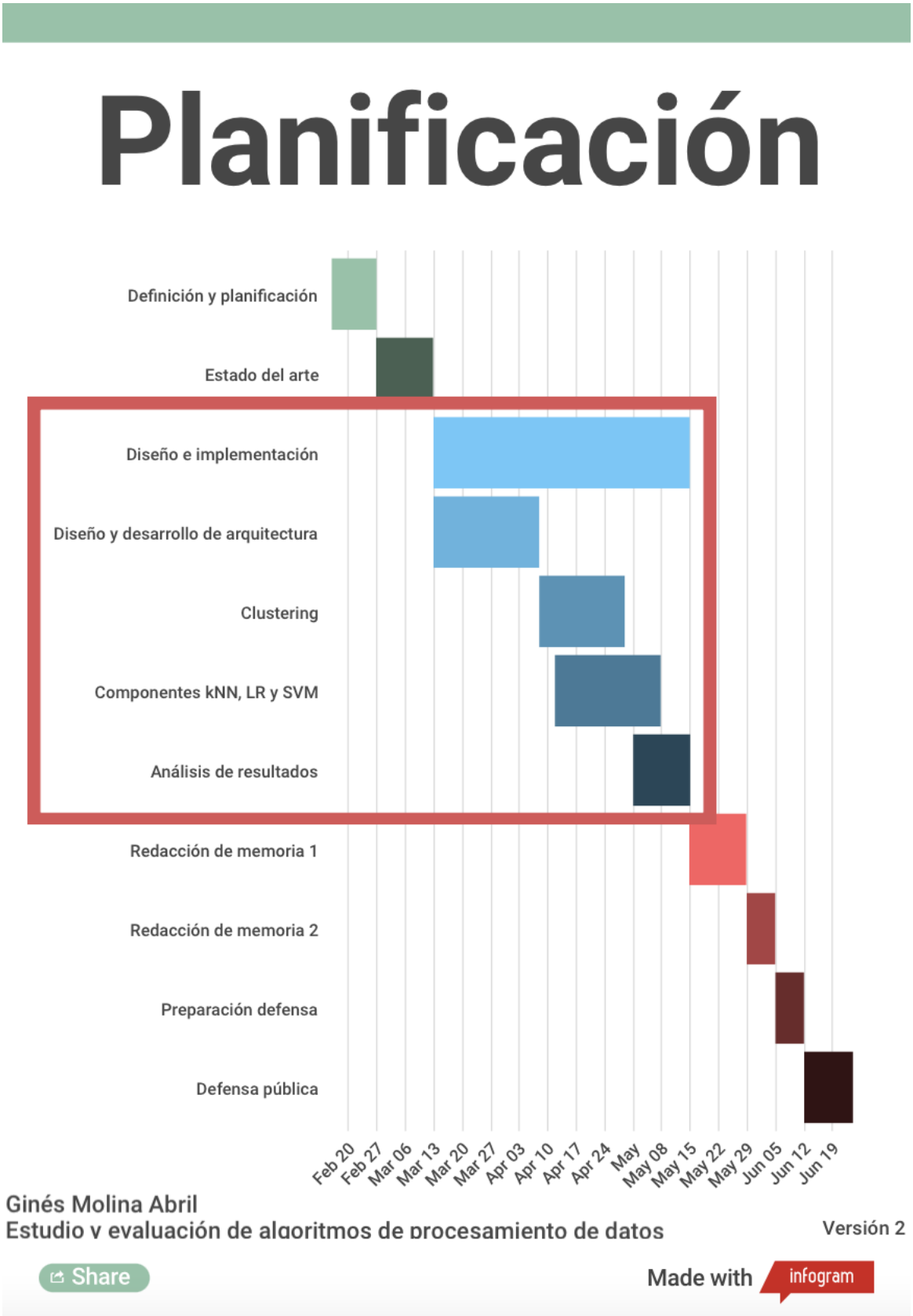


Figura 1.3: Planificación final del proyecto

Capítulo 2

Estado del arte

2.1. Contexto e historia

Ya desde la civilización griega se mostraba la inquietud del ser humano por saber dónde se encontraba y se utilizaban distintas técnicas y herramientas como la posición de los astros o el astrolabio para que, por ejemplo, ayudara en la navegación marítima [54]. El mundo avanza y durante la Segunda Guerra Mundial, por motivos bélicos, se desarrollan sistemas de navegación como GEE (desarrollado por las fuerzas armadas británicas) y LORAN (desarrollado por las fuerzas armadas estadounidenses) [48].

Con las primeras puestas en órbita de los primeros satélites, a partir de los años 60 se desarrollan los sistemas de posicionamiento satelitales norteamericanos TRANSIT, Timation y, por último, Navstar-GPS (origen del actual GPS). La antigua unión soviética desarrolló el sistema GLONASS en 1982, la República Popular China lanza en el 2000 su primer satélite de BeiDou y la Unión Europea comienza el desarrollo de GALILEO en 2003. Gracias al Global Navigation Satellite System (GNSS) la localización en exteriores queda cubierta en la mayoría de escenarios [23].

Estos sistemas de posicionamiento nacen de la necesidad de estas herramientas y llegado a este nivel de desarrollo podemos considerarlas imprescindibles en nuestras vidas ya que no podemos imaginar servicios y productos sin el comúnmente llamado GPS. Aunque estos sistemas cuentan con ciertas dificultades en entornos interiores por problemas de atenuación de la señal recibida dentro de los edificios. Por ello, surgen los sistemas de posicionamiento en interiores (IPS) que, a diferencia de los sistemas de posicionamiento globales (GNSS), no cuentan con un sistema único y se nutren de tecnologías diferentes que se compatibilicen entre sí para ofrecer soluciones fiables y robustas. En el año 2000, Microsoft desarrolla RADAR, que

es el primer sistema de posicionamiento en interiores que aprovecha señales Wi-Fi y abre el camino para posteriores investigaciones y desarrollos usando la técnica de fingerprinting con señales de radio frecuencia, aprovechando las redes e infraestructuras ya existentes [31]. Uno de los problemas principales de los sistemas de posicionamiento en interiores es el costo asociado a los dispositivos de la infraestructura. Por ello, tecnologías como Bluetooth Low Energy (BLE) o Wi-Fi son soluciones asequibles y utilizadas en el desarrollo de numerosas investigaciones.

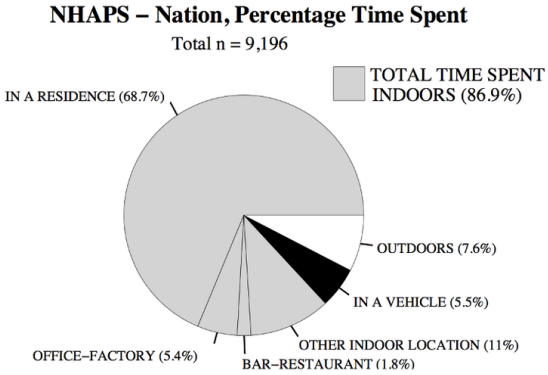


Figura 2.1: Estudio de la NHAPS muestra que los estadounidenses pasan el 86,9 % del tiempo en espacios interiores, mas otro 5,5 % dentro de un vehículo.

Pero, ¿por qué tendría sentido el posicionamiento en entornos interiores? Básicamente porque somos una especie de interior. Según una encuentra del The National Human Activity Pattern Survey (NHAPS) [52], los norteamericanos pasan casi un 90 % de su tiempo en interiores según se puede ver en la figura 2.1. También se puede comprobar dónde pasan el tiempo los encuestados del estudio de la NHAPS a lo largo de 24 horas en la figura 2.2.

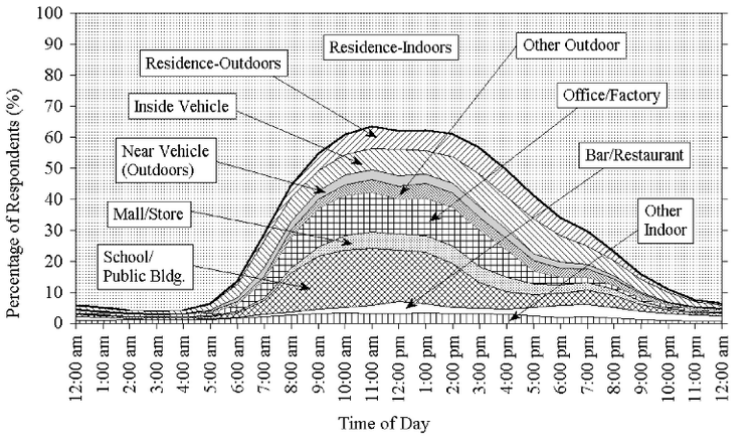


Figura 2.2: Lugares donde pasan el tiempo encuestados por la NHAPS a lo largo de 24 horas.

2.2. Tecnologías y Sistema de Posicionamiento en Interiores

Podemos aprovecharnos de vivir en un mundo en que los dispositivos móviles están más que establecidos y existen infraestructuras de comunicaciones ya instaladas en los edificios ya que no existe ningún estándar en cuanto a tecnología a utilizar. De ahí nace el interés en los sistemas de posicionamiento inalámbrico en interiores utilizando ondas de radio, campos magnéticos, señales acústicas o cualquier otra información que puede ser recogida mediante dispositivos electrónicos o sensores, como por ejemplo a través de señales ópticas. En ambos casos en los que sea o no sea necesario el diseño o inclusión de nuevos dispositivos o sensores en la infraestructura, el sistema en conjunto será llamado Sistema de Posicionamiento en Interiores (IPS) [9].

Los servicios y aplicaciones que se nutren de la utilización de los sistemas de localización son conocidos como Location Based Services (LBS) y tienen que ver sobre todo con el ámbito industrial, el transporte, la salud y la logística [49]. La mayoría de implementaciones actuales usan tecnologías de radio: usando ondas de banda estrecha (BLE y Wi-Fi) u ondas de banda ancha (Ultra Wide Band - UWB) [30]. Según la aplicación requerida se podrá evaluar y diseñar un sistema de posicionamiento en interiores teniendo en cuenta diferentes tecnologías atendiendo al coste, flexibilidad, complejidad, precisión y compatibilidad necesaria para realizar las tareas necesarias que se ajusten más al entorno heterogéneo.

A continuación, se pasan a describir las diferentes técnicas existentes para el cálculo de la posición en interiores y posteriormente se analizarán las ventajas e inconvenientes de varias de las tecnologías que pueden ser utilizadas para desempeñar estas tareas.

2.2.1. Técnicas para cálculo de la posición

Al tratar con infraestructuras inalámbricas, existen diferentes técnicas que pueden ser clasificadas según el tipo de información intercambiada entre el emisor y el receptor. Esta información de la señal puede ser: el tiempo de llegada o Time of Arrival (ToA), la diferencia entre tiempos de llegada o Time Difference of Arrival (TDoA), el ángulo de llegada de la señal o Angle of Arrival (AoA) y el nivel de potencia recibida o Received Signal Strength (RSS). Con esta información y alguna de las técnicas comentadas en apartados posteriores se trata de mejorar la precisión de la estimación de la localización.

ToA es la técnica más precisa que ofrece la posibilidad de eliminar los efectos del multitrayecto en interiores, aunque es la más compleja de implementar y requiere un ejercicio de alto coste de sincronización de todos los dispositivos. La técnica TDoA tiene la ventaja de que sólo necesita sincronizar las estaciones base. Sus puntos negativos son que no elimina los efectos del multitrayecto y necesita cierto conocimiento previo o contexto para afinar la posición. La técnica RSS es fácil de implementar en comparación con las técnicas que utilizan el ángulo de llegada o la diferencia de tiempo de llegada y no necesita hardware especializado en la estación móvil. Como desventaja principal es que la existencia de obstáculos puede atenuar la señal [3].

A continuación, se muestra una figura resumen 2.3 con las características importantes en los sistemas IPS y sus tecnologías asociadas, posteriormente se analizaran aquellas más importantes para este proyecto.

System	Technique	Methods	Accuracy	Calculated at		Scalability	Complexity	Cost	Typical Environment
				Mobile device	Infrastructure				
GPS	Trilateration	ToA	6m–10m	√		Low	High	High	Indoor/ Outdoor
Wi-Fi	Proximity Trilateration Angulation Scene anal	Cell-ID TOA TDOA AOA RSSI	10-100 m (Prox.) 1-10 m	√ RSSI	√ TOA TDOA AOA Prox.	High	Low	low	Indoor
UWB	Trilateration Angulation	RSSI TOA, TDOA, AOA	1cm-1 m		√	Low	Low	high	Indoor
Bluetooth	Proximity Scene anal. Trilateration	Cell-ID RSSI TOA	1-5 m	√ TOA RSSI		High	Low	high	Indoor/ Outdoor
RFID	Proximity Trilateration Scene anal	Cell-ID RSSI	1-5 m	√	√	Medium	Low	low	Indoor
Zigbee	Proximity Trilateration	Cell-ID RSSI	1-10 m	√RSSI	√Prox.	Low	Low	medium	High
Infrared	Proximity Trilateration	Cell-ID TOA	1cm-5m	√TOA	√Prox.	Low	Low	medium	High
Ultrasound	Trilateration	TOA, TDOA	1cm-1m		√	Low	Low	high	Indoor
FM	Trilateration	RSSI fingerprint	2m–4m		√	Low	Low	Low	Indoor/ Outdoor
Video scene analysis	Scene anal. Angulation	Computer vision	1cm-1m	√	√	Low	High	high	Indoor
Barcodes	Proximity Angulation	Pattern recognitin	1-10 m	√	√	Medium	Low	high	Indoor
Sensor networks	Proximity Trilateration	Cell-ID RSSI	10 cm-1 m		√	Medium	High	medium	Indoor/ Outdoor
INS, PNS	Dead reckoning		1-10 m	√		High	Low	low	Indoor

Figura 2.3: Tabla resumen con las características importantes en los sistemas IPS y sus tecnologías asociadas.

2.2.1.1. Fingerprinting

La técnica de fingerprinting obtiene una representación de los dispositivos en una zona de cobertura determinada basándose en los datos obtenidos de RSSI de los distintos puntos de acceso y generando una distribución de probabilidad de los valores RSSI para cada ubicación física determinada [38]. Consiste en dos fases diferenciadas: la fase de calibración o entrenamiento y la fase de ubicación.

1. En la fase de calibración o entrenamiento se construye una base de datos con el mapa de las huellas llamado mapa de potencias (radiomap) que recoge muestras de RSSI en todos los puntos de la cuadrícula de todos los puntos de acceso con el objetivo que sirva de patrón posteriormente.

2. En la segunda fase de ubicación se estima la posición del dispositivo receptor a partir de los mapas de potencias creados en la fase anterior utilizando, por ejemplo, el método de vecino más cercano (KNN), el algoritmo Support Vector Machine (SVM) o aplicando Markov.

Las siguientes figuras (figura 2.4 y figura 2.5) del artículo [37] serían un ejemplo que contextualizan ambas fases aplicando fingerprinting a señales Wi-Fi.

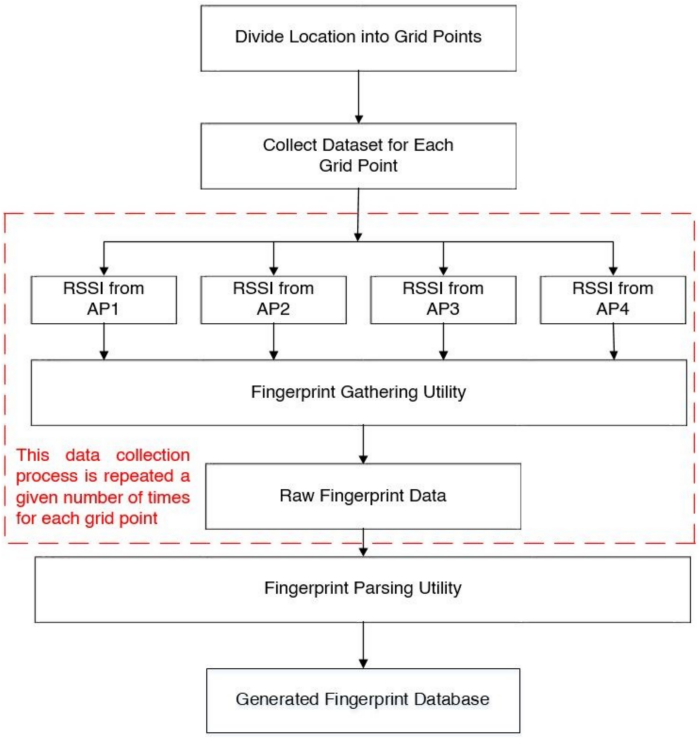


Figura 2.4: Ejemplo de flow-chart del método fingerprinting usando Wi-Fi.

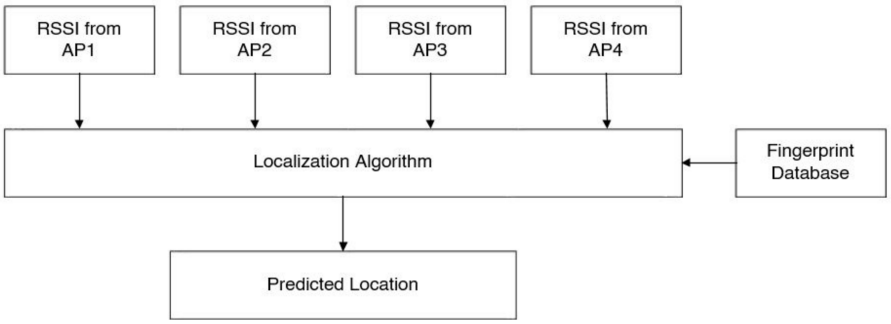


Figura 2.5: Ejemplo del proceso de estimación de la localización con Wi-Fi fingerprinting.

2.2.1.2. Triangulación de potencias

Esta técnica utiliza de nuevo múltiples puntos de referencia para obtener la localización del dispositivo. Con las potencias recibidas de cada emisor se crea un sistema de ecuaciones. Se obtienen los puntos de triangulación en base a la resolución de este sistema de ecuaciones. A continuación, cada punto de triangulación se considera un vértice del triángulo. Una vez se calculan todos los triángulos posibles teniendo en cuenta todos los emisores y se comparan sus áreas, se obtiene un centro del triángulo con el área mínima (o a través del cálculo del Minimum Mean Square Error MMSE) que sirve para finalmente estimar la localización del receptor. Se puede ver un ejemplo claro de la técnica en la figura 2.6 del artículo fuente [24].

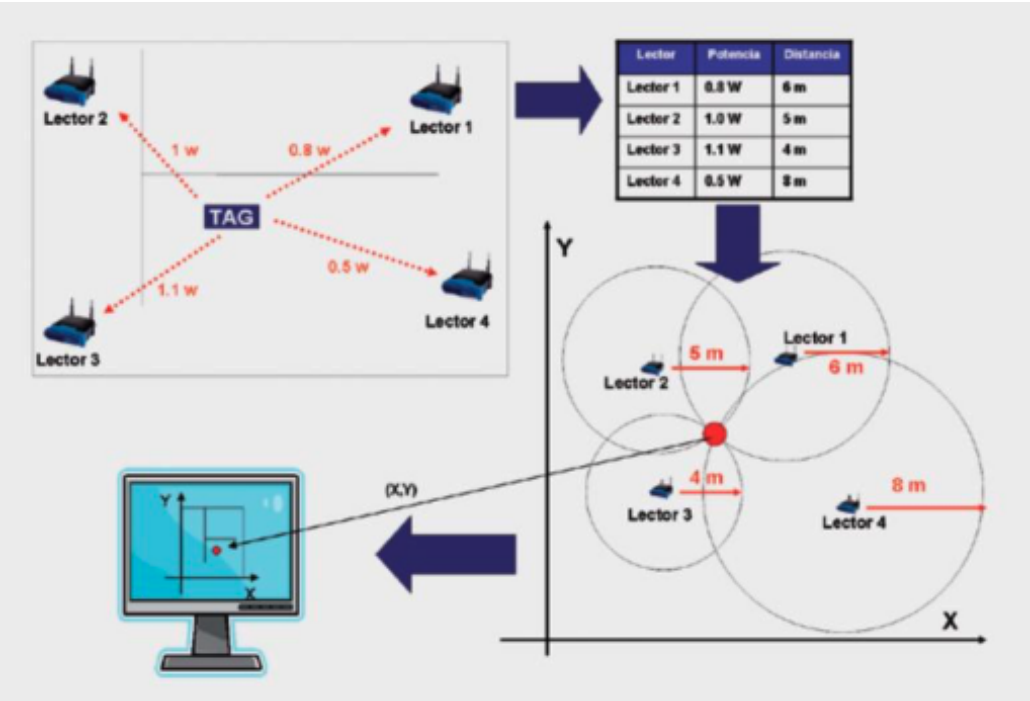


Figura 2.6: Triangulación de potencias para, por ejemplo, un sistema RTLS.

2.2.1.3. Angulación

La técnicas de Angulación Angle of Arrival (AoA) o Angle of Destination (AoD) requieren antenas adicionales con capacidad para medir los ángulos que forma la dirección de propagación de la onda incidente y una determinada orientación o dirección de referencia. De esta manera, con la capacidad de los sensores de medir los ángulos se puede establecer una posición estimada como aparece en la figura 2.7 y la fuente consultada es el artículo [3].

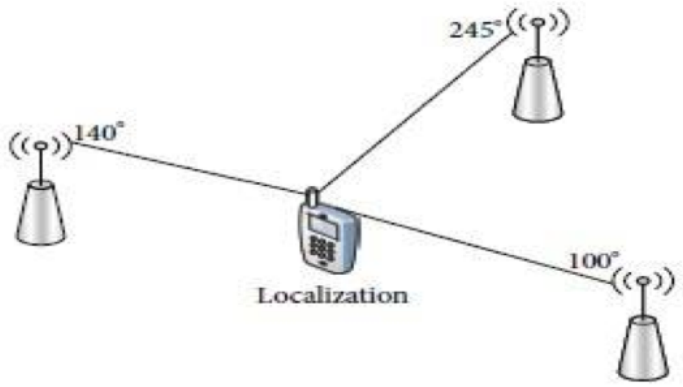


Figura 2.7: Técnica de posicionamiento según el ángulo del receptor.

Un caso de aplicación para Bluetooth AoD sería como la figura 2.8, de nuevo, y la fuente consultada es el artículo [3].

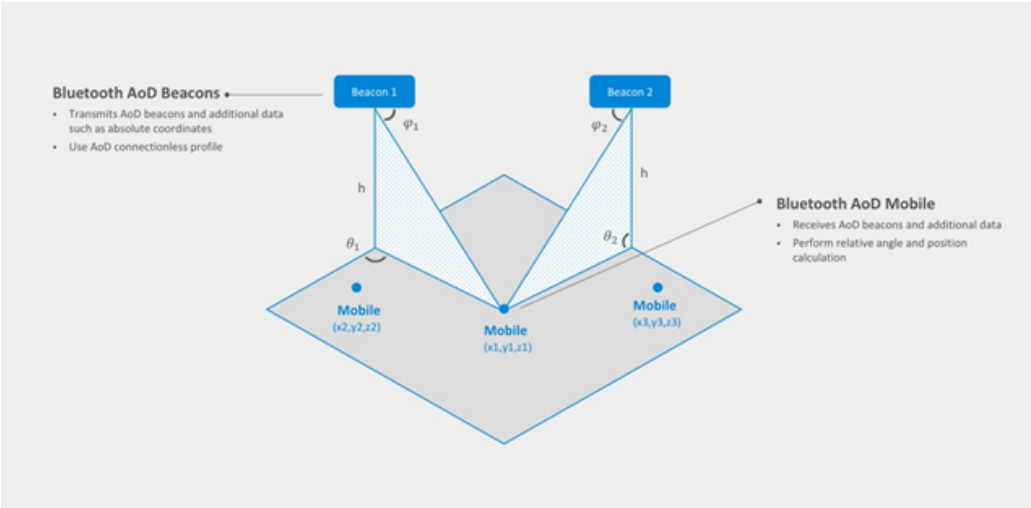


Figura 2.8: Ejemplo de AoD utilizando Bluetooth.

Este tipo de técnicas pueden ser interesantes además al ser utilizado en IPS basados en señales luminosas [10].

2.2.2. Tecnologías

2.2.2.1. Wi-Fi

El protocolo 802.11, o mas comúnmente llamado Wi-Fi, sigue un modelo centralizado. Esto quiere decir que cada red consta de uno o varios puntos de acceso o APs y un grupo de clientes conectados a ellos. Cada uno de los puntos de acceso emite periódicamente una baliza o beacon, para que así los clientes puedan acceder a las redes disponibles en su entorno. Se trata de la tecnología más popular para el posicionamiento en interiores y es uno de los principales focos de investigación, sobre todo recientemente con el avance en Machine Learning. Generalmente se cuenta con una red de puntos de acceso de la que se obtienen las medidas RSSI a través de señales Wi-Fi de los distintos puntos de acceso con el objetivo de determinar la distancia del dispositivo a cada uno de ellos y poder finalmente estimar la posición del dispositivo mediante la aplicación de la técnica de trilateración [53].

Existen ciertas ventajas al utilizar Wi-Fi como tecnología en estos sistemas IPS. La primera de ellas es que existen puntos de acceso Wi-Fi por toda la ciudad, en centros comerciales, en tiendas, restaurantes y lugares públicos. Estos sistemas de localización indoor Wi-Fi que utilizan técnicas de fingerprinting están catalogados dentro de los sistemas que no requieren infraestructura siempre que se hagan uso de las redes WLAN ya creadas. Por tanto, el coste de implementación de la infraestructura es bajo y esto permite que sea una alternativa muy atractiva para la investigación. Además, se cuenta con un ancho de banda suficiente, velocidad aceptable, expansión y flexibilidad de la red y ya viene integrado en teléfonos inteligentes. Esto quiere decir que se contaría con 3000 millones de potenciales usuarios a nivel global [43].

Aunque Wi-Fi tiene un alcance mayor que BLE, es un alcance limitado. Se ve muy afectado por la atenuación de la señal si hay objetos sólidos con materiales no conductores entre el beacon y el receptor. Al trabajar con señales se tiene en cuenta que su naturaleza no es lineal, por lo que las medidas siempre quedarán afectadas por obstáculos que se puedan encontrar en el entorno además de estar obligado a contar con atenuación por la distancia del AP al usuario. Para las nuevas definiciones de Wi-Fi lo más importante es ganar más alcance, no más velocidad [44].

Otra de sus ventajas es que cualquier dispositivo con Wi-Fi activado está constantemente buscando puntos de acceso y esto hace que automáticamente el sistema pueda detectar su posición sin necesidad de autenticación o permisos de conexión. Otro de los problemas es el multitrayecto, provocado por los fenómenos de reflexión, difracción y dispersión, por la misma

naturaleza de la onda. A esto habría que sumar el efecto que podrían provocar otras posibles transmisiones que estuvieran utilizando el mismo espectro de frecuencias [29]. Uno de los objetivos o enfoques a adoptar que se podría plantear con la utilización de esta tecnología para IPS es la resolución de un problema de clasificación y posterior minimización de errores para mejora de la estimación [53].

2.2.2.2. BLE

Bluetooth Low Energy ofrece tecnología de bajo costo, alta eficiencia energética, fácil de implementar, con funcionamiento independientemente de la red, con menos interferencias y fácil de integrar en sistemas de Internet of Things (IoT) [21]. Al tratarse nuevamente de una tecnología de radio usando ondas de banda estrecha el modo de proceder sería muy similar a la utilización de Wi-Fi, utilizando triangulación se podría averiguar su localización. En este caso y como se ha comentado previamente, el alcance es menor que con Wi-Fi, por lo que para cubrir un mismo área se necesitarán más dispositivos. El error cometido, aunque se puede afinar un poco más, podría estar en torno a 1,5 metros [53].

Los elementos básicos que interactúan con la tecnología son llamados BLE beacons y BLE scanners. El estándar Bluetooth define dos tipos de transmisión: transmisiones de datos y transmisiones de publicidad o advertising. La transmisión de datos utiliza canales específicos de radio frecuencia (37 canales dedicados) para comunicar entre los dispositivos conectados entre sí. La primera fase de transmisión de publicidad o advertising utiliza 3 canales físicos dedicados de radio frecuencia para el descubrimiento de dispositivos, inicializar una conexión o difundir información mostrando las intenciones de conexión [15].

Con un modo parecido de funcionamiento a Wi-Fi, el dispositivo BLE beacon trata de difundir periódicamente los beacons con señales de radio que son escaneadas y capturadas por el dispositivo BLE scanner. En este caso, es necesario que el dispositivo BLE scanner esté en modo de recibimiento o escucha monitorizando los canales de advertising o publicidad: canal 37, canal 38 y canal 39 (seleccionados para minimizar la interferencia con los canales Wi-Fi) [51]. De esta manera, cada dispositivo BLE scanner obtiene un valor Received Signal Strength Indicator (RSSI) medido en dBm que posteriormente es utilizado para posicionar el dispositivo midiendo la distancia entre el emisor y el receptor. Una vez se establecen las coordenadas físicas en las que se encuentran los emisores y se comparan con los valores de señal y el RSSI, se estará en posición de estimar la posición aproximada del receptor. En este caso generalmente se usa el método de trilateración como método de mejora de precisión [1].

2.2.2.3. UWB

La banda ultra-ancha o Ultra-Wide Band es una tecnología nueva que no requiere el uso de la comunicación tradicional y transmite datos en un amplio rango de ancho de banda. UWB parece una buena tecnología para la localización en interiores, permite altísimas velocidades de transmisión y rangos elevados. Una de sus ventajas es que su transmisión puede considerarse ruido de fondo para otras tecnologías inalámbricas, por lo que en teoría, puede utilizar cualquier espectro sin interferir en otras comunicaciones. Utiliza una potencia de transmisión baja, lo que hace que el consumo también sea bajo. Otra de las ventajas respecto a Wi-Fi y BLE es que no le afectan los problemas de multi-trayecto. Es considerada tecnología de corto alcance con precisión de 20 a 30 centímetros pero su hardware es mas costoso. Además, se trata de una tecnología aún no regulada para usar libremente a escala masiva, al contrario de las tecnologías anteriores. Se trataría de una tecnología ideal que busque precisión y robustez ante cambios del entorno [4]. Sus posibilidades de futuro están algo más próximos a ser el sustituto del USB para conseguir un ecosistema de dispositivos conectados entre sí sin necesidad de cableado [29].

2.3. Aplicaciones de posicionamiento en interiores

La figura 2.9 es un resumen con algunas de las aplicaciones de las tecnologías de localización y sus métodos o técnicas obtenida del artículo [4].

Application domains	Indoor localization technologies	Accuracy
Asset tracking	RFID, Infrared (IR), Barcode Scanning	Very high
Location-based advertising	Bluetooth beacons (iBeacon)	Low
Shopping assisstance system	Bluetooth, WiFi	High
Museum, fairs, airports, guided tours	Wi-Fi fingerprinting ,Bluetooth LE	Medium
School, university campus	Wi-Fi fingerprinting, Wi-Fi TOA/TDOA	Medium
Hospitals, healthcare, Ambient Assistant Living (AAL)	Wi-Fi fingerprinting, RFID, IR	High
Emergency response and rescue management	Ultrasound, dead eckoning	Medium
Robotics	Dead recko ning,infrared, UWB,Vision sensors	Very high
Indoor sports	Bluetooth LE	Very high
Smart home	ZigBee	High
Augmented reality	LLA Markers(barcodes), INS	High

Figura 2.9: Algunas de las aplicaciones de las tecnologías y métodos de localización.

2.3.1. Soluciones de proximidad

Este tipo de soluciones tienen un bajo coste pues sólo es necesario configurar una infraestructura sencilla (BLE por ejemplo) y establecer el comportamiento de la aplicación o servicio que tendría conforme se acerca a los diferentes puntos de interés o áreas de cobertura. Aunque se traten de

aplicaciones muy básicas, para ciertos contextos es mas que suficiente. Otra posibilidad es la integración de estos sistemas como servicio dentro de sistemas IoT globales o de domótica ya presentes en espacios públicos para así aprovechar la infraestructura ya creada [33].

2.3.1.1. Punto de interés

Este tipo de aplicaciones cuentan con un Point of Interest (PoI) usado para descubrir dispositivos cercanos y, por ejemplo, enviar información de marketing personalizada basándose en la ubicación del usuario. La ubicación del usuario queda determinada dependiendo de la cercanía al punto de interés [1].

Los datos obtenidos de los puntos de interés serán de vital importancia para, por ejemplo, entender el comportamiento del usuario y así ofrecer información oportuna y precisa según cada usuario. También se podría alimentar el sistema con información pública de los usuarios en redes sociales o en base a sus preferencias para poder ofrecer un mejor servicio. Otras de las muchas posibles aplicaciones pueden ser, por ejemplo: un sistema de información, visualización de instalaciones, visitas de usuarios en zonas urbanas, mapas a gran escala o sistemas de navegación y etiquetado de objetos y espacios en plataformas de realidad aumentada [7].

2.3.1.2. Detección de artículos

Otra de las aplicaciones que pone en práctica una solución de proximidad es la búsqueda de elementos u objetos para, por ejemplo, localizar artículos extraviados. No necesita una red costosa y se puede hacer uso de la infraestructura ya creada con la adquisición de algunos dispositivos más asequibles que sean etiquetados [1].

2.3.2. Soluciones de posicionamiento

2.3.2.1. Sistemas de posicionamiento en tiempo real RTLS

Existe la necesidad de localizar personas y objetos en interiores en tiempo real con un alto grado de precisión. Un sistema RTLS es una combinación de hardware y software que se utiliza para determinar y proporcionar continuamente la posición en tiempo real de activos y recursos equipados con dispositivos diseñados para operar con el sistema [20].

Una de las aplicaciones de RTLS es la gestión de activos para facilitar su localización o rastreo. Una configuración ideal está basada en servidor-puntos de acceso, que registran los valores de todos los dispositivos etiquetados a lo largo del área de cobertura enviando estos valores al servidor central. Finalmente, el motor de ubicación utilizará esta información

del servidor, los valores RSSI y la posición conocida de los puntos de acceso para ubicar el dispositivo etiquetado. Otra de las posibles aplicaciones es la de optimizar tiempos de funcionamiento de los equipos etiquetados en un gran almacén, mejorando la eficiencia, prediciendo tiempos de mantenimiento y reduciendo tiempos de inactividad o zonas de mayor confluencia de activos donde pueden dar lugar a cuellos de botella o una alta concentración de dispositivos [1].

Otra de las aplicaciones es en el ámbito de la salud. En el ámbito sanitario podrían existir limitaciones como la prevención de interferencias en el funcionamiento de cualquier equipo médico y retos como maximizar la duración de la batería [35]. Según el ámbito de aplicación pueden surgir limitaciones o problemas que deberán ser tomados en cuenta en el diseño del sistema RTLS desde el principio.

2.3.2.2. Sistemas de posicionamiento en interiores IPS - Wayfinding

Este tipo de sistemas son sistemas de información utilizados en la mayoría de ocasiones para guiar a las personas a través de ambientes físicos y mejorar su comprensión y experiencia del espacio [8], o para comprender el comportamiento de los usuarios. Las balizas o transmisores están montados en lugares fijos del área de cobertura, a diferencia de los sistemas RTLS que pueden permanecer en continuo movimiento [1].

Como se ha visto anteriormente, pueden utilizarse técnicas como el fingerprinting o la triangulación de potencias para que, con información de los valores de RSSI, se logre posicionar al usuario. Uno de los objetivos que se plantean es la integración en entornos heterogéneos en el que además se tengan en cuenta varios niveles o pisos en edificios. Habrá que tener en cuenta que podrán existir distintos tipos de interferencias y que cada entorno será distinto, por lo que se usará métodos heurísticos para mejorar las técnicas principales. De entre todos estos métodos, el más sencillo de implementar asegurando un correcto funcionamiento y una mayor precisión en entornos de interiores es el basado en la trilateración de RSS combinada con estimación mejorada utilizando un enfoque de minimización de errores, como por ejemplo, el análisis del escenario previo que permita realizar un mapa de potencias [11].

2.4. Localización en interiores mediante Wi-Fi RSS Fingerprinting

Numerosos estudios realizados demuestran que las principales ventajas de la localización en interiores mediante señales Wi-Fi a través de RSS-Fingerprinting son el elevado número de características de cada fingerprint y la buena precisión conseguida incluso utilizando un algoritmo simple como KNN. En cuanto a limitaciones o desventajas hay que tener en cuenta que este tipo de sistemas tienen una frecuencia de refresco baja y que los valores RSSI pueden estar afectados por otras señales, obstáculos o incluso por el numero de personas en el entorno (pues se obtendrían distintos patrones de intensidad de la señal).

Como ya se ha visto anteriormente con la técnica de fingerprinting, la localización de un dispositivo con un mapa de potencias, a grandes rasgos, consiste en hacer coincidir cada medida con alguno de estos valores previos. El primer sistema que lleva a cabo este tipo de técnicas con Wi-Fi es RADAR, basado en un método similar al KNN (precisión de 3 a 5 metros). Mas tarde se aplican algoritmos como SVM, consiguiendo una tasa de error mucho más baja [37].

Los algoritmos más usados están basados en redes neuronales artificiales o algunas de sus variantes (CNN, DNN, BPNN, capsnet, RNN, DSCP, MLP, QKMMCC, NN), que resultan el 61 % del total de trabajos analizados según [5]. Por la naturaleza de la señal Wi-Fi, la red neuronal es una buena elección al ser adecuada para funciones no lineales. En uno de los papers [27] se consigue una medida de error medio muy buena con 0,11 metros usando una red neuronal profunda con Wi-Fi mmWave.

2.5. Localización en interiores mediante BLE

Como se ha comentado anteriormente, BLE es una tecnología apta para integrar en sistemas IoT y convivir con otras tecnologías. Las ventajas que aporta la tecnología es un valor extra a los análisis de entornos híbridos y heterogéneos. Además del citado artículo [15] en el que se hace un estudio empírico de un sistema de localización a nivel de habitación es interesante descubrir si los procesos y desarrollos aplicados para sistemas y datasets Wi-Fi RSS-fingerprinting tienen el mismo efecto con este tipo de sistemas.

Se parte también de los estudios [53] y [34]. Estos dos estudios comparan características de ambos sistemas de Wi-Fi y BLE además de aplicar el desarrollo de un algoritmo a ambos de ellos para comprobar diferencias. En concreto, en [53] la implementación del posicionamiento

en interiores utilizando señales Wi-Fi obtiene mejores resultados y muestra como el sistema BLE en interiores parece no tan robusto y depende mucho del entorno.

2.6. Modelado

¿Por qué clustering? Un problema de los mapas de radio muy densos (con muchas muestras) es que algunos modelos no son eficientes, ya que se tiene en cuenta todo el entorno.

Con clustering se puede aplicar el principio de "Divide y vencerás"

2.6.1. Algoritmos de clustering

Con clustering se puede aplicar el principio de "Divide y vencerás". Un problema de los mapas de radio densos o con muchas muestras es que algunos modelos no son eficientes, ya que se debe tener en cuenta todo el entorno. Esto quiere decir que se cuenta con un problema global en un entorno heterogéneo y se decide dividir el problema en otros más fáciles de operar y aquí es donde los algoritmos de clustering nos facilitará la labor. Existen muchas investigaciones orientadas a la aplicación de algoritmos supervisados con datos etiquetados. En el caso concreto de este proyecto en el que se cuenta con un dataset de validación para realizar la validación "a ciegas" y sin etiquetas, será importante aplicar previamente un algoritmo de clustering como por ejemplo k-means o DBSCAN. La función de estos algoritmos previos será la de encontrar diferencias entre las muestras, organizar por puntos en común y aportar información importante para las predicciones posteriores [40]. Estas organizaciones en subconjuntos podrán ser usadas para, según unos datos de entrenamiento, asignar un subconjunto a cada registro objetivo, ya sea de test o validación y poder predecir en base a estos subconjuntos una zona o habitación.

K-Means es un algoritmo que intenta encontrar una partición de las muestras en K agrupaciones. El mejor valor de K para que la clasificación separe lo mejor posible los grupos no se conoce a priori, dependerá de cada contexto. Cada una de las muestras pertenecerá a uno de estos K grupos dependiendo de qué centroide sea el más cercano. Una posible utilización de K-Means en el sistema puede ser el de dividir en distintos valores de K para, posteriormente, analizar cuál de ellos es el que obtiene un mejor resultado dependiendo del edificio, planta o clúster.

Ya que, al tratarse de aprendizaje no supervisado, no se cuenta con las etiquetas de clases correctas que ayudaría a medir la calidad del algoritmo, se deberá hacer uso de las métricas intrínsecas junto a las visualizaciones. Lo ideal sería utilizar una métrica en la que también pueda utilizar la distancia intergrupo. K-means no tiene un buen rendimiento si la disposicion de los puntos de las variables tienen formas complejas. Con el uso de K-Means asumimos que

trabajamos con celdas de Voronoi y que los clusters son circulares. Para los casos en los que la disposición de los puntos de las variables tienen formas complejas se pòdrían utilizar algoritmos basados en densidad.

DBSCAN es un algoritmo de clustering basado en densidades que marca como outliers los valores que superan un límite de frontera establecido. En este caso DBSCAN no hace suposiciones de que cuenta con clusters esféricos. Se trata de un algoritmo de clustering muy utilizado y relativamente eficiente y no necesita especificar el número de clusters deseado, al contrario que con K-Means. Tiene la capacidad de encontrar clusters con formas geométricas arbitrarias pero habrá que hacer especial énfasis con el tipo de distancia que se aplica (comparándolo con KNN) para que no dé lugar a mostrar valores no deseados [6]. Las variables importantes a establecer en el algorimo son "eps" y "min_samples". Según estos valores se trata de identificar qué puntos son "puntos núcleo", cuáles son "puntos borde" y cuáles "puntos de ruido" para en base a sus disposiciones se puedan marcar los límites de los grupos. En este caso se le da más importancia a este proceso de comprobar la disposición de los puntos núcleo y puntos borde para establecer unas fronteras que tengan sentido.

La aportación importante de estos modelos en el sistema es la de encontrar subconjuntos de registros para que se puedan diferenciar respecto al sistema general y poder aplicar uno u otro método para averiguar el espacio físico de cada registro a predecir [14]. Se debe tener en cuenta que previamente no se conoce el espacio ni como están distribuídas las habitaciones.

2.6.2. Algoritmo KNN

El algoritmo KNN o k vecinos más cercanos es un algoritmo de clasificación supervisado con una lógica muy simple basado en instancia (Lazy Learning). Esto quiere decir que el algoritmo no aprende explícitamente un modelo, sino que memoriza las instancias de entrenamiento que son utilizadas como "conocimiento" para la fase de predicción. El objetivo del algoritmo es el de clasificar o estimar un registro o valor en función de sus valores más cercanos. El parámetro más importante es la cantidad de valores cercanos "n_neighbors" que tendrá en cuenta el algoritmo. Teniendo en cuenta que existen numerosos tipos de medidas como la euclídea o la Manhattan, este será otro de los hiperparámetros a definir a través del parámetro "metric" o "p". El parámetro "p" por defecto tiene el valor 2 y más específicamente es equivalente a la métrica de distancia euclídea estándar. El valor de "p" igual a 1 sería equivalente a usar distancia de Manhattan. KNN se podría utilizar para problemas de clasificación y de regresión.

El proceso es muy simple, una vez se le pasan los valores de entrenamiento, calcula la distancia entre cada registro y el resto dependiendo de la métrica de distancia. A continuación, ordena esas distancias de manera ascendente y elige los valores K superiores. Finalmente asigna la clase más frecuente de los valores anteriores al registro.

KNN tiene entre sus ventajas que es muy simple. Además, es un algoritmo no paramétrico que lo hace muy útil en problemas con datos no lineales, como es el caso. No tiene una fase de entrenamiento especializada, tan sólo almacena los resultados en memoria. Esto puede ser contraproducente y computacionalmente caro en términos de tiempo y almacenamiento si se trata de un dataset grande. Otra de las consideraciones importantes a tener en cuenta es que el algoritmo es sensible al escalado y propenso a fallos si no se escalan los datos, es decir, si las distintas variables tienen distintas escalas puede llegar a producir errores y tiene sentido al tratarse de distancias.

2.6.3. Regresión logística multinomial

La regresión logística multinomial es un algoritmo de clasificación supervisado "1-vs-All" muy simple y eficiente que se utiliza cuando la variable objetivo es categórica y se desea encontrar la clase con probabilidad máxima, es decir, el valor de la variable objetivo con probabilidad máxima en base a un conjunto de datos que tienen variables de entrada numéricas. Además se diferencia del modelo de regresión lineal en que se utiliza la estimación de probabilidad máxima, esto proporciona una puntuación de probabilidad para cada clase [41].

Este algoritmo resulta muy interesante si se contara en el sistema con una única variable categórica objetivo y se aplicara como solución a un problema de clasificación multiclase. Esta variable objetivo podría ser el identificador del edificio, la planta, la habitación o la zona. En el caso de este proyecto puede tratarse de un identificador único que diferencie cada una de las combinaciones entre las distintas variables objetivo. La implementación de la clasificación multiclase sigue las mismas ideas que la clasificación binaria, pero en este caso tenemos más de dos clases. El algoritmo también es llamado clasificador de máxima entropía (MaxEnt). Los parámetros interesantes a evaluar son el "penalty" con valores "L2","L1" y "None" y "C" o inverso de la fuerza de regularización, cuyos valores más pequeños especifican una regularización más fuerte.

2.6.4. Algoritmo SVM

Otro de los algoritmos de aprendizaje supervisado que se puede utilizar es el algoritmo SVM o Máquinas de Vector Soporte. Aunque inicialmente se desarrolló en la década de los 90 como un método de clasificación binaria, su aplicación se ha extendido a problemas de clasificación múltiple y regresión [22]. Se partirá de nuevo con un conjunto de datos de entrenamiento etiquetados por clases. El objetivo del algoritmo es el de tratar de separar de manera óptima las diferentes clases mediante un espacio lo más amplio posible para que así los nuevos datos puedan ser clasificados exitosamente sin dar lugar a error o duda. El algoritmo busca el hiperplano que tenga la máxima distancia con los puntos que estén más cercanos, así los puntos etiquetados con una categoría estarán a un lado del hiperplano y el resto al otro lado [19]. En un espacio de dos dimensiones, el hiperplano es un subespacio de 1 dimensión (una recta) y en espacio tridimensional es un subespacio de dos dimensiones.

A diferencia del "Maximal Margin Classifier" en el que se basa SVM y también comparte el concepto del hiperplano, SVM prefiere no separar perfectamente las clases para que el modelo no sea tan sensible a los nuevos datos y así dotarlo de mayor robusted y mayor capacidad predictiva al no depender tanto de las muestras de entrenamiento (menos problemas de overfitting). Si se emplea un kernel lineal, el clasificador Support Vector Classifier (SVC) controla la regularización a través del hiperparámetro C. Otra de las ventajas es que SVM utiliza la optimización convexa, lo que garantiza que el resultado sea un mínimo global. Para los casos en los que no haya manera de encontrar un hiperplano que permita separar dos clases, se dice que las clases no son linealmente separables y se recurre al "truco del núcleo" que consiste en utilizar una dimensión nueva en la que podamos encontrar ese hiperplano que separa las clases.

2.6.5. Otros algoritmos y arquitecturas

2.6.5.1. Arquitecturas Transfer Learning

Teniendo en cuenta que, en vista general, este proyecto busca un conjunto de modelos, procesos y prácticas que ayuden a obtener la posición de dispositivos en entornos heterogéneos, una vez podemos tener agrupados un serie de conjuntos en diferentes clases el siguiente paso es poder tomar un enfoque de reducción del error de posicionamiento en distintas bases de datos. Este enfoque se puede lograr desarrollando un modelo de aprendizaje de transferencia que tenga en cuenta distintos datasets con sus propias particularidades y con valores no etiquetados previamente.

Los algoritmos de transferencia por aprendizaje consisten en seleccionar una red o arquitectura neuronal exitosa y ya entrenada y con unos pesos de la red neuronal ya calculados que sirvan para nuestro problema. Hay que tener en cuenta que habrá que adaptar y ajustar la arquitectura neuronal al problema y al realizar algunas iteraciones, se podrá incluso reducir el error de posicionamiento hasta un 25 %, consiguiendo eliminar valores atípicos, como muestra una de las investigaciones previas. El algoritmo propuesto en este caso consiste en tres bucles: el primero crea los modelos pre-entrenados y codifica los datos, el segundo realiza el entrenamiento del modelo de Transfer Learning y el último afina el modelo final para cada despliegue individual [25].

2.6.5.2. Arquitecturas Incremental Learning

Partiendo del enfoque anterior con arquitecturas Transfer Learning nace el interés por aplicar algoritmos Incremental Learning que puedan tener la habilidad de adaptación continua del modelo basado en un flujo de datos que llegan constantemente, como por ejemplo de un sistema IoT. Este enfoque podría ir aprendiendo según pasa el tiempo y se recibe nuevos datos con, posiblemente, nuevas características. Una de las limitaciones es la limitación de memoria, por lo que es importante establecer una ventana temporal apropiada que pueda capturar el concept drift [16].

Anteriormente se ha tomado este enfoque en investigaciones referentes al aprendizaje incremental auto-supervisado para la localización de fuentes sonoras en entornos interiores complejos y heterogéneos obteniendo muy buenos resultados [28].

2.6.5.3. Técnicas ensemble models

Por último, se puede tomar otro enfoque que parta de los resultados anteriores y de los distintos modelos y arquitecturas para descubrir qué modelo es bueno para qué subconjuntos de puntos de prueba y así añadir este componente en el pipeline, es decir, construir un clasificador intermedio para clasificar qué punto es el que mejor se ajusta a un modelo anteriormente aplicado siguiendo cada uno de los enfoques.

Como se muestra en la investigación [36], se consiguen muy buenos resultados comparado con ensemble models tradicionales. En ese caso en concreto el componente cuenta con 3 etapas: la inicialización del nuevo ensemble model, entrenamiento del nuevo ensemble model (iteración y cálculo de errores) y finalmente la etapa de predicción en la que se utiliza el mejor modelo para cada conjunto.

Capítulo 3

Diseño, modelado y desarrollo del proyecto

3.1. Entorno y descripción del dataset

Para este proyecto se utiliza el dataset UJIIndoorLoc creado por la Universitat Jaume I en el que se han recogido los datos recibidos de 25 dispositivos Android. Este dataset está dividido en 4 ficheros, dos de ellos de entrenamiento y dos de ellos de test. Existe un fichero mas de validación que cuenta con otras mediciones de potencias a ser utilizadas para predecir "a ciegas" y validar los resultados con una fuente externa ajena al sistema. Los ficheros de entrenamiento están separados, por un lado con las variables con las potencias de los puntos de acceso, y por otro lado las variables "y" con las coordenadas (en metros), identificador de edificio y planta. Estos valores serán utilizadas para averiguar el posicionamiento en interiores de cada uno de los registros. Cada uno de los registros representa una medición o una huella Wi-Fi asociados a un set de potencias de señal recibidas en formato RSSI (dBm) por cada uno de los puntos de acceso desde 520 es diferentes. Posteriormente, veremos que los valores se toman desde -104 dBm a 0 dBm, donde -104 sería un valor más débil y 0 un valor más fuerte. El valor 100 representa que no hay señal, es decir, que no se capta el dispositivo.

Los ficheros de entrenamiento cuentan con 19.937 registros, mientras que los de tests cuentan con 1.111 registros. Estos registros han sido recolectados en 3 edificios diferentes enumerados como edificio 0, edificio 1 y edificio 2. Los edificios 0 y 1 comprenden cada uno 4 plantas numeradas desde el 0 al 3. El edificio 3 comprende 5 plantas numeradas del 0 al 4.

3.2. Deuda técnica en proyectos ML

Según el tipo de empresa o equipo de investigación, puede llegar a pasar que se deba proporcionar una solución basada en la puesta en producción de manera flexible de los distintos modelos o arquitecturas de modelos que vengan a proporcionar una solución en la predicción del posicionamiento en interiores. Con la adopción e integración de la IA por las empresas en sus productos y servicios y según se explica en [50], se ha encontrado con la necesidad de encontrar un proceso simplificado *end-to-end* que fuera sostenible al escalar y que sea, al mismo tiempo, monitorizado y *trackeado* para poder llevar a producción la arquitectura de modelos planteada. En desarrollo de software es común hablar de 'deuda técnica' y en este caso también afectaría en cuanto al coste de mantenimiento en sistemas de ML del mundo real [46] tal y como muestra la figura 3.1.

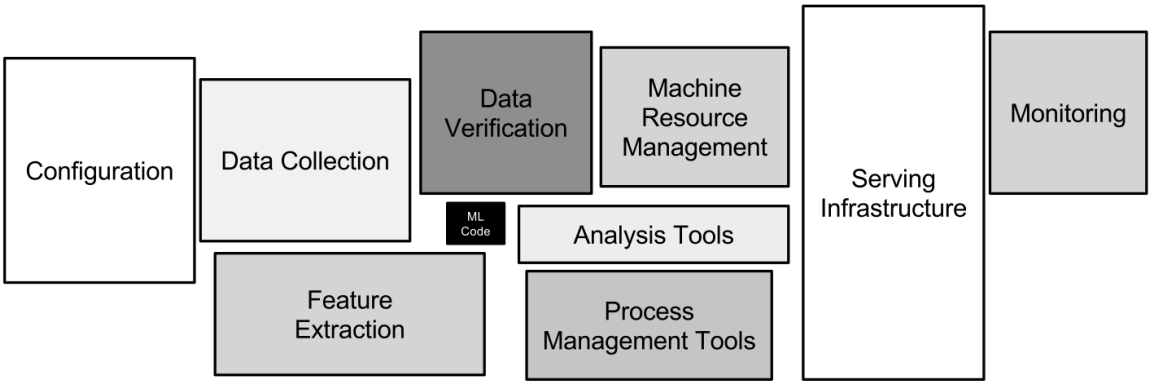


Figura 3.1: Deuda técnica en sistemas de ML - Sólo una pequeña parte de los sistemas de ML del mundo real está compuesta por el código de ML. La infraestructura necesaria es amplia y compleja.

En este tipo de desarrollos e investigaciones que se plantean se presentan una serie de retos que involucran a varios responsables y *skateholders* de la empresa o del equipo investigador al trabajar en la puesta en marcha de modelos de Machine Learning en producción. Según [45], existen las siguientes limitaciones:

1. **Calidad del dato**. Se deberá tener en cuenta la calidad del dato en el sistema desde un punto de vista flexible. Sin duda se deberá contar con una etapa o componentes en el sistema en que se puedan realizar un correcto pre-procesado de los datos para verificar que el modelo obtiene datos bien definidos y se filtren posibles valores incorrectos. Para lograr mitigar este problema en este proyecto se plantea una etapa previa que se encarga de reportar las características del dataset que se ha cargado, limpiar los datos y realizar las transformaciones oportunas (posteriormente se tratará en profundidad esta etapa). Es

imprescindible dotar de flexibilidad en el pre-procesado o dotar de una buena calidad del dato al sistema si se entiende que se trata de un sistema flexible que puede obtener en cualquier momento algún dato inesperado.

2. **Descomposición del modelo**. Por el mismo motivo que en el punto anterior, al tratarse de un sistema flexible que cuenta con un proceso de varias iteraciones, se entiende que el modelo se adaptará a estos cambios que puedan producirse con el paso del tiempo. Esta limitación se ha tenido en cuenta en la etapa de modelado y arquitectura del sistema. Los modelos deberán ser entrenados teniendo en cuenta que pueden existir cambios en cuanto a las características de los datos y, por este motivo, deberá dotarse al componente de cierta flexibilidad para que los cambios no afecten al global del sistema y pueda ser gestionado de manera parcial o local en cada modelo. Además del reporte mencionado anteriormente, con esta solución planteada en este proyecto se logra poder trabajar de manera flexible en cada modelo de la arquitectura de manera independiente para atender a cualquier cambio que haya que realizar sobre él.

3. **Localidad de datos**. Es importante dotar al sistema de cierta estabilidad respecto a las modificaciones que sufre el dataset a lo largo del proceso. Será muy importante poder definir y etiquetar cada grupo de datos o archivos que se han ido procesando de manera independiente por cada componente teniendo en cuenta que posteriormente será utilizado por otro componente. Estos datos deberán ser fiables y correctamente almacenados en cada punto del proceso en que se haya realizado una modificación de ellos, para así poder tener un mapa global de las transformaciones que se llevan a cabo y poder trabajar, posteriormente y de manera independiente sobre cada grupo de archivos o agrupación de datos.

3.2.1. Enfoque MLOps

Este nuevo enfoque que comparte tareas con el perfil DevOps en desarrollo de software viene a completar las piezas que faltan en cuanto al conocimiento de las herramientas de ML que se pueden utilizar pensando en llevar a producción un pipeline orientado a este tipo de productos y servicios de ML [18].

Está claro que hay tareas que comparten principios y buenas prácticas entre ambos roles. Cabe destacar por ejemplo que en el caso del perfil MLOps tiene que tener en cuenta, además de las limitaciones anteriormente expuestas, un Almacén de metadatos con los datasets utilizados. En el proyecto se aplican estos principios para finalmente encajarlo en algo similar al diagrama siguiente:

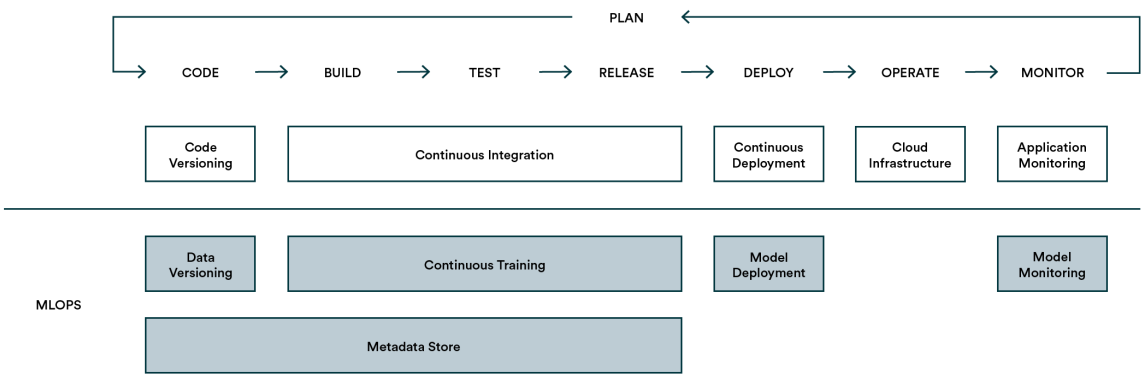


Figura 3.2: Comparativa de tareas entre un DevOps y MLOps

Las Herramientas de Ploomber y MLFlow ayudarán en dotar al pipeline de la flexibilidad y reportabilidad que se necesita. También cabe destacar que en este tipo de perfiles, además de tener en cuenta la Integración Contínua (CI) y la Implementación Contínua (CD) en el ciclo de vida del flujo de datos de ML, se debe tener en cuenta el Entrenamiento Contínuo o Continuos Training (CT) de la arquitectura de modelos planteada.

3.3. Tecnologías y sistemas

El entorno técnico de trabajo se compone de una serie de herramientas enfocadas en distintas etapas y con el objetivo de cumplir algún requisito o paliar algún problema detectado según el workflow comentado anteriormente.

1. **Anaconda, Python y librerías asociadas**. En primer lugar se decide utilizar el entorno de desarrollo de Anaconda. Esto nos va a permitir poder establecer un entorno reproducible en cuanto a las librerías instaladas en nuestro equipo localmente y nos permitirá poder reproducir nuestro pipeline en cualquier lugar sin problemas de librerías. En primer lugar, ha sido necesario la creación de un repositorio **GIT** en GitHub:

   github.com/yinx89/ips-algorithms-study.

   Se utiliza **Python 3.7** porque es la versión más estable con el framework Ploomber que comentaremos posteriormente. Python tiene la versatilidad del lenguaje y permite trabajar con librerías muy potentes como **scikit-learn**, **pandas** o **numpy**. Se puede seguir el proceso de instalación en el Anexo C .

2. **Ploomber y MLFlow**. Ploomber es un framework clave en la realización de este proyecto. Se encargará de orquestar el desarrollo del pipeline en cada una de las etapas y de facilitar la labor de definición de las distintas etapas, trackeo de etapas, escalado a nivel de arquitectura de modelos, independencia de ejecución, paralelismo y almacenamiento

de objetos y reportes agrupados por experimentos [12]. Además, permite el desarrollo ágil del pipeline por su gran abanico de posibilidades que ofrece al funcionar perfectamente con otras herramientas como MLflow o los Jupyter Notebooks. Se trata de un framework open-source muy joven pero con un futuro interesante en cuanto a sus posibilidades y su facilidad de uso.

Por otro lado, MLFlow es una plataforma también open-source encargada de gestionar el ciclo de vida del proceso de ML. Es la encargada de mostrar la experimentación con los modelos, ofrecer reproducibilidad de los modelos y su despliegue ágil en producción asignando distintas etiquetas si queremos diferenciar modelos en producción o en pruebas. Cuenta con un registro central de modelos que nos permite tener una instancia en el Cloud en la que se puede apuntar a través de una URL endpoint para registrar los experimentos y los distintos eventos. Finalmente, se podrá almacenar en el cloud los experimentos, artefactos y modelos. También desde ahí se pueden obtener las predicciones atendiendo a las distintas versiones de los modelos registrados. Para servir modelos con MLFlow se necesita un servidor de seguimiento en ejecución y un identificador de ejecución del modelo. Se puede seguir una guía de instalación y uso en el Anexo D.

Así es como se vería un ejemplo del fichero .yml con uno de los componentes:

```
meta:
  extract_upstream: False
tasks:
  # STEP 1 : GET DATA & EVALUATE

  - source: tasks/1_get_data_evaluate.ipynb
    name: Get data & evaluate
    product:
      nb: products/1_get_data_evaluate/get_data_evaluate.ipynb
    params:
      input_path_training_coord: "{{root}}/input_data/UJI1_trncrd.csv"
      input_path_training_rss: "{{root}}/input_data/UJI1_trnrss.csv"
      input_path_test_coord: "{{root}}/input_data/UJI1_tstcrd.csv"
      input_path_test_rss: "{{root}}/input_data/UJI1_tstrss.csv"
```

Y así quedaría el conjunto de tecnologías utilizadas:

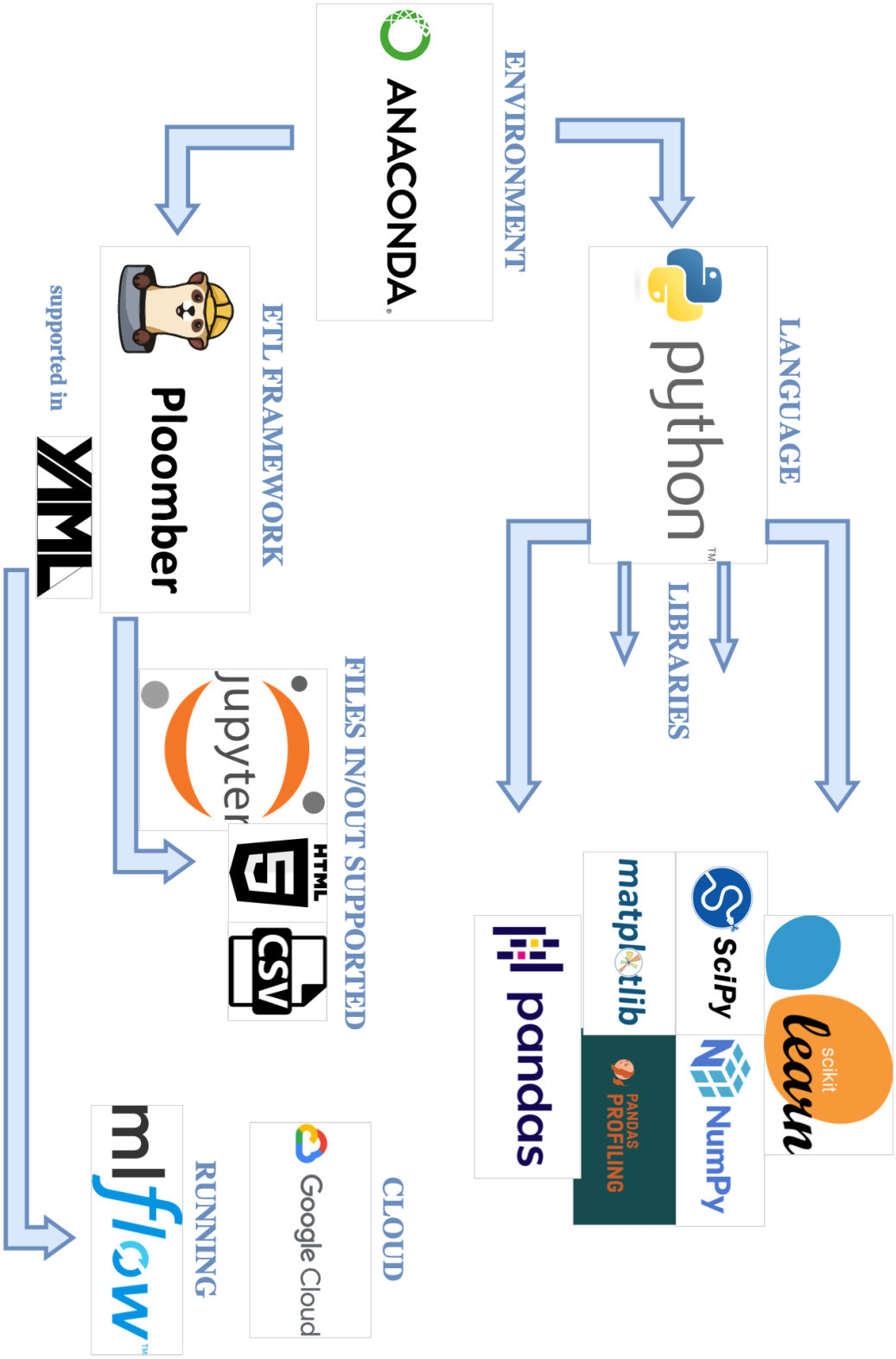


Figura 3.3: Diagrama global de tecnologías

3.4. Flujo de datos

Siguiendo los apartados anteriores, a continuación, se definirá cómo estas herramientas interactúan entre sí y cómo se lleva a cabo cada ejecución del pipeline teniendo en cuenta qué valores son los de entrada y cuáles los de salida según se muestra en el diagrama global del Anexo F .

En primer lugar, se crea una carpeta input data que será la encargada de alojar los archivos de entrada. En este punto del proyecto se definen estos valores teniendo en cuenta unos ficheros concretos de un dataset concreto, pero en otros casos el valor de entrada puede ser el nombre de la carpeta y el código podría obtener distintos datasets cargados en esa carpeta automáticamente. Estos cuatro ficheros corresponden a la información que se ha detallado en el apartado 3.1. La ruta a estos ficheros son entrada de las tareas 1, 2, 3, y 4. Las salidas o productos de cada una de estas tareas o componentes serán guardados en la carpeta **products**. También se pasan como variables de entrada los valores que corresponden al endpoint de MLFlow así como el nombre del experimento que se quiere registrar y los archivos utilizados una vez han realizado las transformaciones y las labores de limpieza de los datos en componentes posteriores.

1. **Obtener y evaluar el conjunto de datos**. En este primer componente se obtienen los datos de una carpeta de entrada, se registra un resumen del dataset y se muestran los valores de cada columnas en un archivo de texto .log. Esto nos ayuda a encontrar por cada dataset, qué columnas tienen según qué valores y así posteriormente tomar decisiones de pre-procesado en base a esta información. Si estos valores cambiaran porque el fichero es actualizado, se generaría un nuevo reporte, por lo que podemos tener siempre bajo control qué valores se tiene en cada momento de ejecución del pipeline. Un ejemplo del formato del texto a modo de resumen se puede encontrar en el Anexo E.

   Como se puede ver en el reporte anterior, habrá casos en los que encontramos que para un cierto punto de acceso puede no tener valores relevantes de señal y podemos desechar esta columna. Esta etapa es importante para encontrar información que previamente no contábamos con ella al visualizar la tabla de valores, y junto con la etapa siguiente, será importante para definir las tareas de pre-procesado y limpieza de los datos.
   También se puede observar en esta etapa que en el dataset de entrenamiento existe un desbalance respecto al número de registros según los distintos edificios. Se observan más de 8.000 registros para el edificio 2 y casi 5.000 para los edificios 0 y 1, esto sin duda afectará a los resultados.

2. **Reportar dataset**. En esta etapa se llama a la función **ProfileReport** de la librería pandas profiling. El resultado se guarda en un fichero html además del notebook de Jupyter. Este componente presenta un análisis exploratorio profundo de los datos con una extensión de la funcionalidad describe() de pandas en un informe HTML interactivo. En el Anexo G se puede encontrar un ejemplo de uno de los reportes que genera el componente en esta etapa respecto al archivo de coordenadas de training.

3. **Visualizar dataset**. En esta etapa, junto con la información obtenida en etapas anteriores, el objetivo es profundizar en la viualización de las variables del dataset. Para empezar, se realiza un Scatter Matrix de los atributos y se observa que no hay relaciones lineales entre ninguno de esos atributos:

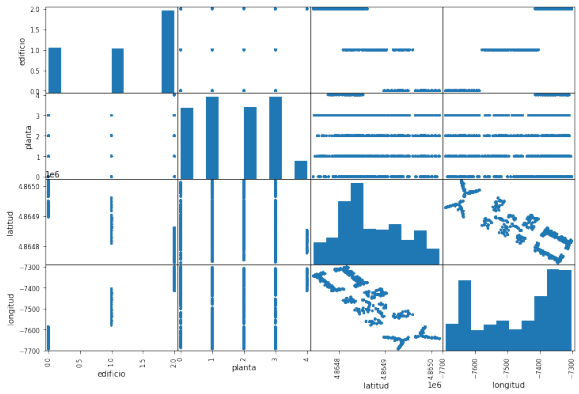


Figura 3.4: Scatter Matrix de los atributos

A continuación, se muestra el mapa de correlaciones entre puntos de acceso. Se pueden observar pequeños focos de correlación entre distintos puntos de acceso, aunque la mayoría de variables no tienen correlación y no afectarán en la clasificación. Aún así, en este punto se advierte de la maldición de la dimensionalidad y se evitará reduciendo el número de características del dataset con PCA en posteriores etapas:

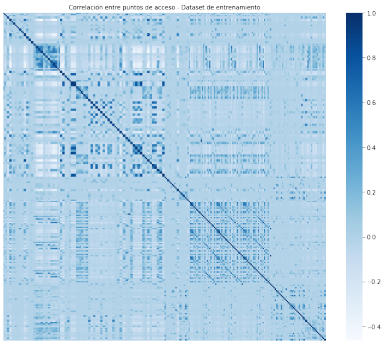


Figura 3.5: Correlaciones - WAPs

4. **Limpieza de datos**. El objetivo de este punto es la limpieza de los datos y el agrupamiento o pre-procesamiento para facilitar el paso de modelado posterior y poder obtener grupos de datos a través de sus ficheros .csv y .npy. Como en los pasos anteriores, se cargan los ficheros de entrada de la carpeta input data. A continuación, se encuentran los valores de señal igual a 100 y se establecen al nuevo valor -105 (ya que sabemos que el valor mínimo es -104). De esta manera pasamos los valores de potencia a la escala positiva sumando a todos los valores 105. Esto se realiza tanto para el grupo de training como para el de test. En el siguiente paso se eliminan todos los registros que muestran en todos los valores de potencia igual a 0. Esto quiere decir que no ha sido registrado por ningún punto de acceso. También se eliminan todas las variables o columnas que representan un punto de acceso que no hayan captado algún dispositivo, de nuevo, tanto para el dataset de training como para el de test. Ahora se decide obtener una unica variable capaz de representar las coordenadas de altura, latitud y longitud. Esta variable toma el nombre de "localización", y será muy importante para agrupar posteriormente por zonas en cada edificio o planta y para utilizar en los algoritmos ya que, como se ha indicado anteriormente, por ejemplo para el caso de la Regresión logística multiclase es necesario una única variable categórica objetivo. Esta variable obtiene 933 puntos de acceso únicos. En este punto también se convierten las variables edificio y planta a tipo String. Se agrupan subconjuntos de datos segun identificador de edificio y planta, se normalizan los valores RSSI y se aplica PCA. Con PCA se busca reducir el conjunto de atributos a un nuevo conjunto con menos dimensiones maximizando la varianza explicada tras la proyección. Se elige PCA ya que, a diferencia de t-SNE, PCA no da lugar a resultados diferentes con los mismos datos.

   Para cada agrupación, se guarda un fichero .csv y otro fichero .npy con los valores WAP tanto para el dataset de training como para el dataset de test. El código asociado a este apartado se puede encontrar en /tasks/4_clean_data.ipynb.

5. **Clustering K-Means**. En este componente el objetivo es el de utilizar los subgrupos generados anteriormente según el identificador de edificio, la planta y la informacion de los WAPs del dataset de training para aplicar K-Means, obtener y almacenar los resultados en ficheros .npy y mostrar gráficas sobre distintas métricas: la inercia, el coeficiente de silueta medio, integridad, homogeneidad, v-measure y el coeficiente de silueta medio, como se puede mostrar en la figura 3.6 para el identificador de edificio 0. Este paso previo en la arquitectura de los modelos en paralelo es importante y se utilizará para dividir por espacios o zonas. No se ha visto importante utilizar Clustering para determinar el identificador del edificio o la planta ya que se obtienen muy buenos resultados en la etapa posterior sin necesidad de aplicar Clustering. Por lo que, esta etapa tiene la

función determinada de preparar los subgrupos o clusters para conseguir posteriormente encontrar un modelo que prediga correctamente según los subgrupos generados en este punto con distintos valores K. El código asociado a este apartado se puede encontrar en /tasks/5_clustering.ipynb.

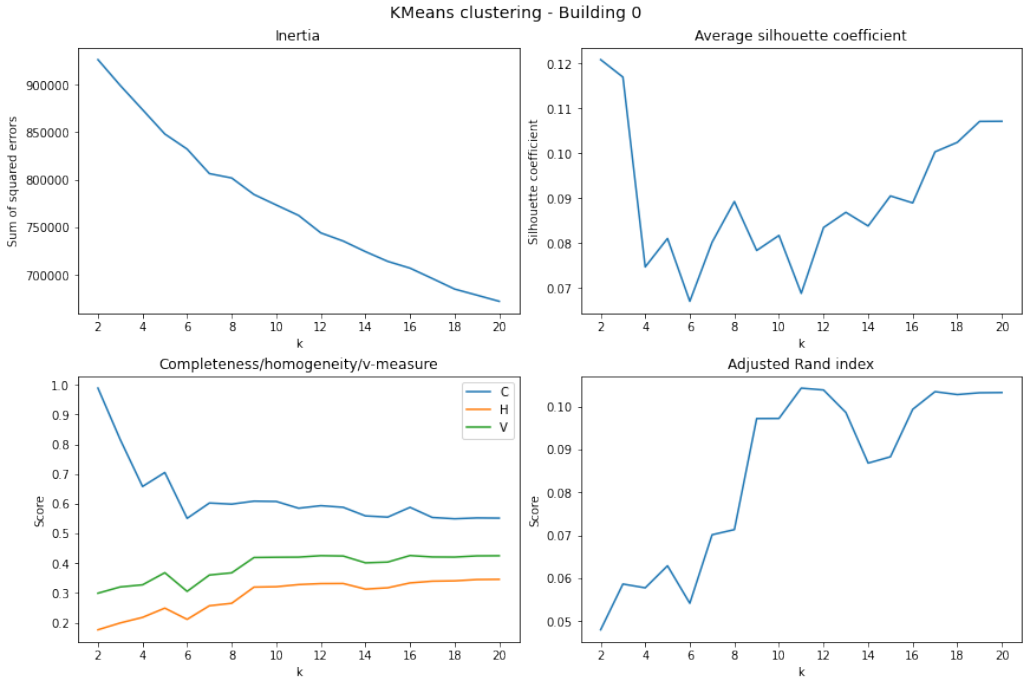


Figura 3.6: KMeans Clustering in Building 0

Para los pasos siguientes el flujo de datos se divide y el pipeline ejecuta las distintas arquitecturas en paralelo. Este es el diagrama resultante que se puede autogenerar con Ploomber:

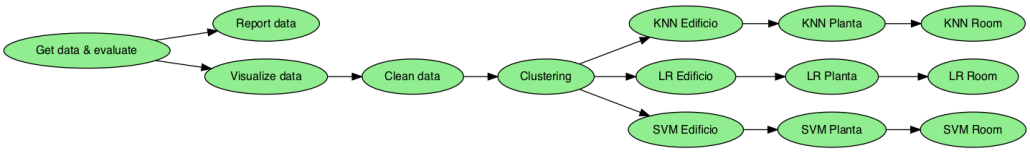


Figura 3.7: Ploomber plot - Autogenerado

Como se puede ver en el diagrama, las distintas arquitecturas de modelos se enfocan primero en la predicción del identificador de edificio. A continuación, se enfocan en entrenar un modelo para predecir la planta según cada edificio y por último, se entrena un modelo para predecir la localización teniendo en cuenta los clusters anteriores según distintos valores de K. Todo este proceso será trackeado con MLFlow, por lo que será muy fácil comparar entre modelos y elegir cuál es el llevado a producción.

6. **Predicción de Edificio**. En este apartado se rescatan los ficheros de training sin agrupar, el fichero con las potencias de test y los ficheros .npy con los objetos Numpy con las potencias de entrenamiento y test ya en formato preparado para entrenar el algoritmo. Como se ha indicado anteriormente, se comienza trackeando el proceso con MLFlow atendiendo a los parámetros de entrada con la URL, el identificador del experimento y el nombre del experimiento.

   A continuación, se llama a la función que, al mismo que tiempo que entrena el modelo (con "kd_tree" para KNN, con random_state=1 para LR y con kernel="linear" y random_state=1 para SVM), establece los hiperparámetros objetivo para finalmente buscar el mejor modelo en base a ellos. Esta función devuelve el mejor estimador (que será almacenado en una instancia de Google Cloud junto al resto de métricas), los mejores parámetros, y las puntuaciones.

   Finalmente, se guarda el mejor modelo en la ruta de productos y se registra en MLFlow acorde al nombre del experimento. Ahora se calculan las distintas métricas que serán también registradas en MLFlow por cada ejecución: av_balanced_acc (media de las puntuaciones del dataset de entrenamiento), varianza (media de las varianzas del dataset de entrenamiento), test_accuracy (calculado una vez obtenidas las predicciones en base al mejor modelo entrenado), n_neighbors y p para KNN, penalty-l2, penalti-None y C para LR y C para SVM.

7. **Predicción de Planta**. En este apartado se rescatan los ficheros de training agrupados por identificador de edificio, los ficheros con las potencias de test agrupados también por identificador de edificio y los ficheros .npy con los objetos Numpy con las potencias ya en formato preparado para entrenar el algoritmo de entrenamiento y test también agrupados por identificador de edificio. Se comienza trackeando el proceso con MLFlow atendiendo a los parámetros de entrada con la URL, el identificador del experimento y el nombre del experimiento.

   A continuación, se llama a la función que, al mismo tiempo que entrena el modelo (con "kd_tree" para KNN, con random_state=1 para LR y con kernel="linear" y random_state=1 para SVM), establece los hiperparámetros objetivo para finalmente buscar el mejor modelo en base a ellos. La diferencia en este caso es que el proceso se realiza una vez por cada edificio y se pasa cada dataset agrupado por identificador de edificio. Otra de las diferencias es que en este caso se apunta a la variable "planta" como objetivo. Esta función

devuelve el mejor estimador, los mejores parámetros y las puntuaciones.

Finalmente, se guarda el mejor modelo en la ruta de productos y se registra en MLFlow acorde al nombre del experimento. Ahora se calculan las distintas métricas que serán también registradas en MLFlow por cada ejecución: av_balanced_acc (media de las puntuaciones del dataset de entrenamiento agrupado por identificador de edificio), varianza (media de las varianzas del dataset de entrenamiento agrupado por identificador de edificio), test_accuracy (calculado una vez obtenidas las predicciones en base al mejor modelo entrenado), identificador de edificio, n_neighbors y p para KNN, penalty-l2, penalti-None y C para LR y C para SVM.

8. **Predicción de localización**. En este apartado se rescatan los ficheros de training agrupados por identificador de edificio y los objetos Numpy que hacen referencia a los clusters calculados en la etapa anterior con KMeans. Lo interesante de esta etapa es entrenar un modelo para cada valor de K y una serie de sugrupos generados según cada identificador de edificio y encontrar qué valores de K han sido los más exitosos. Se comienza trackeando el proceso con MLFlow atendiendo a los parámetros de entrada con la URL, el identificador del experimento y el nombre del experimiento.

   A continuación, se llama a la función que entrena el modelo (igual que antes con "kd_tree"), "n_neighbors=3" y "p=2" para KNN, "random_state=1" para LR y con kernel="linear", C=0.1 y random_state=1 para SVM. Para el trabajo futuro sin duda que se podría implementar en esta etapa de nuevo la búsqueda de mejores parámetros, esto haría ajustar el mejor modelo con los mejores hiperparámetros. En este caso el proceso se realiza de nuevo una vez por cada edificio y se pasa cada dataset agrupado por identificador de edificio. Esta función devuelve un contador con el número de registros según el valor de K establecido y el contador de registros según las distintas localizaciones en el clúster. Estos valores ayudarán a descubrir si en este caso el modelo recibe datos desbalanceados. También se recibe un dataframe con las puntuaciones obtenidas por cada valor de K, identificador de edificio y planta. Cada uno de los modelos se registran como anteriormente y se registran las métricas para poder identificar cuáles son los valores de planta, identificador de edificio, precisión y K.

   Finalmente, por cada identificador de edificio se obtendrá una matriz de resultados teniendo en cada fila cada una de las plantas del edificio y en cada una de las columnas el valor de K con el que se ha entrenado el modelo. En esta arquitectura, aunque se hayan

ejecutado dos algoritmos, tan sólo es necesario registrar el último, pues tendremos para elegir qué modelo es el mejor, ya que los clusters han sido almacenados en un objeto Numpy.

Cada uno de los componentes anteriores se puede encontrar en la carpeta /tasks. Además se cuenta con un carpeta /utils con los siguientes archivos que sirven para refactorizar las tareas anteriores:

1. **classifierfunctions.py** - Funciones externas de apoyo a la clasificación con el objetivo de refactorizar código y aprovechar las funciones entre los distintos componentes.

2. **clusteringfunctions.py** - Funciones externas de apoyo a las labores de clustering con el objetivo de refactorizar código. Incluyen funciones para dibujar gráficas.

3. **logger.py** - Funciones encargadas de realizar el registro de eventos.

4. **pipeline.png** - Gráfica autogenerada por Ploomber con el pipeline general.

5. **predicciones.ipynb** - Template para realizar predicciones a cada uno de los modelos en producción. Se debe tener en cuenta que antes de realizar las predicciones se deberá enviar los registros de potencias en el formato adecuado, además de eliminar los puntos de acceso que no fueron tomados en cuenta para el entrenamiento y aplicar PCA.

6. **test-list-buckets.py** - Fichero de ayuda para verificar la correcta conexión con Google Cloud Storage.

3.5. Cross-Validation y métricas de éxito

En el proyecto se aplican técnicas de "grid-search" para la búsqueda de los mejores parámetros de los algoritmos y también se utiliza el método de validación cruzada K-fold para la evaluación de rendimiento de los algoritmos. Esto viene a resolver los problemas de fiabilidad por distintos motivos, como por ejemplo: por una distribución desigual de los datos al particionar entre datos de entrenamiento y prueba, por la afectación de valores atípicos o por contar con clases desbalanceadas [26]. El algoritmo es entrenado con unos valores de entrenamiento y posteriormente será validado probando con unos valores de prueba o test. Es importante que el algoritmo sea entrenado y validado para que sus valores sean estadísticamente fiables y lo suficientemente genérico para que se puedan pedir predicciones con un cierto grado de precisión. En el enfoque básico, el conjunto de datos se divide en k conjuntos más pequeños determinado por el usuario y se sigue el siguiente procedimiento o bucle para cada una de los subconjuntos k: por cada

iteración el modelo se entrena con (k-1) subconjuntos dejando el otro subconjunto para validación del modelo, es decir, cada subconjunto es tratado como conjunto de prueba una vez en todo el proceso global. La medida de precisión obtenida mediante validación cruzada es entonces la media de los valores calculados en el bucle [47]. En este proyecto, el valor de K elegido es 10.

Las métricas de precisión o éxito fruto de la clasificación que realizan los algoritmos se obtienen a través de las matrices de confusión resultantes y sus valores: Verdadero Positivo, Verdadero Negativo, Falso Negativo y Falso Positivo, de cada una de las clases atendiendo a los valores reales y los resultados de las predicciones. Por ello, se obtendrá un valor de precisión medio conforme a los datos de entrenamiento y posteriormente se obtendrá un valor de precisión con los datos del dataset de prueba.

Capítulo 4

Resultados y predicciones

En base al diagrama de flujo anterior podemos distinguir los resultados en las predicciones en tres grupos: identificador de edificio, identificador de planta y localizacion.

4.1. Predicciones - Edificio

KNN está basado en instancia de tipo supervisado y lo que hace es comprobar la clase de los k vecinos más cercanos del set de entrenamiento. En este caso, puede pasar que para valores de k bajos, los datos alejados del núcleo de su clase sean distribuidos dentro de otras posibles fronteras de decisión y el algoritmo sea sensible ante datos que se alejen del núcleo de su clase, dándo lugar a una clasificación errónea. También puede pasar lo contrario y que los valores de k sean tan grandes que impida una distribución coherente. En la búsqueda de los mejores hiperparámetros se consigue el mejor modelo con n_neighbors=3 y p=1 (ver capítulo 2.6.2). En este caso se consigue un valor de precisión balanceada media del clasificador del 99.84 % con una varianza de 0.177 para valores de entrenamiento, una precisión de 99.38 % para los datos de testeo y un tiempo de ejecución de 6.3 minutos. Comparándolo con el resto de algoritmos resulta algo elevado, para finalmente dar una precisión de testeo más baja. Como se ha indicado previamente, un mayor número de muestras de entrenamiento afecta en el rendimiento de este algoritmo respecto al resto. Cabe destacar que en este caso la distancia apropiada para el algoritmo es la distancia de Manhattan (p=1) o más bien, la suma de las diferencias absolutas de sus intensidades.

Se puede comprobar en el cuadro 4.1 que los algoritmos LR y SVM obtienen un 100 % de precisión con los datos de prueba en la predicción del edificio, con la diferencia de que la Regresión Logística Multiclase consigue estos fantásticos resultados en casi una cuarta parte del tiempo de ejecución o entrenamiento en el que lo hace SVM. Para SVM se consigue el

mejor modelo con C=0.001 y para LR se consigue el mejor modelo con C=0.5 y penalty=l2. El objetivo en SVM es el de encontrar el equilibrio entre sesgo y varianza para que el modelo no se sobreajuste o quede falto de ajuste. El parámetro C permite controlar la amplitud del márgen o límite de decisión entre los vectores de soporte y, ya que el objetivo de SVM es maximizar este margen y maximizar el número de puntos clasificados correctamente, se busca reducir el valor del parámetro C todo lo que se pueda sin que afecte en la precisión del modelo. Si se buscara un modelo que separara completamente las diferentes clases sin error obtendríamos sobreacondicionamiento y una alta precisión sobre el conjunto de entranamiento que no serviría para otros conjuntos de datos. Este modelo sería un modelo muy sensible a pequeños cambios y SVM busca algo más generalizado. En lo que se conoce como proceso de regularización y para valores más bajos del parámetro C, el modelo se vuelve menos estricto con los datos clasificados erróneamente (se elige un límite de decisión con un margen más grande a expensas de un mayor número de clasificaciones erróneas). Para valores más altos de C el modelo se vuelve más estricto respecto a las penalizaciones por error en la clasificación (se trata de un margen menor porque SVM trata de minimizar el número de valores mal clasificados).

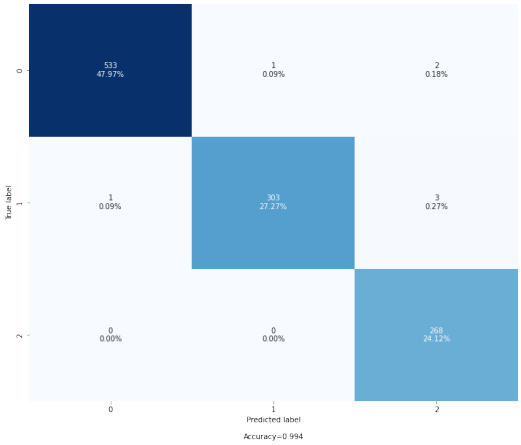
En cuanto a la Regresión Logística Multiclase, por el mismo proceso del algoritmo al basarse en la clasificación binaria, se intuía que iba a ser más rápido que SVM. El entrenamiento es idéntico al procedimiento usado en la Regresión Logística y consiste en minimizar la entropía cruzada (que es la función de error) usando el método del Gradiente Descendente. Las funciones de activación son no lineales y las fronteras de decisión calculadas son lineales. En este caso, los grupos son fácilmente separables, es decir, para el algoritmo poder determinar si se trata de un edificio u otro, los datos que corresponden a cada edificio deberán ser separables, que ya sabemos que lo son según se ha podido representar en etapas anteriores. Para este tipo de predicciones con subconjuntos separables la regresión logística multiclase se trata de la mejor opción ante otros algoritmos más complejos o que requieren más tiempo.

Cuadro 4.1: Predicciones de Edificio

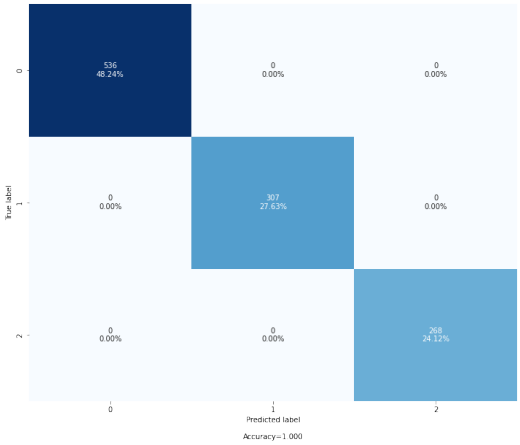
	KNN	LR	SVM
av_balanced_acc	99.84 %	99.96 %	99.95 %
variance	0.177	0.091	0.091
test_accuracy	99.38 %	100 %	100 %
t_train_ex	6.3 min	0.85 min	2.9 min
t_pred	10.351 ms	1.35 ms	1.62 ms

En la figura 4.1 se muestran las matrices de confusión sin grandes diferencias entre sí. Estos

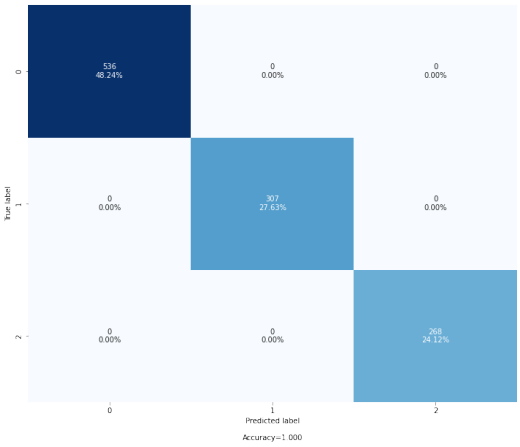
resultados para el algoritmo KNN son ligeramente superiores a los obtenidos en el artículo [47]. Respecto a SVM, los valores del artículo anterior son mucho más bajos (62 % frente al 100 %).



(a) Matriz de confusión - KNN



(b) Matriz de confusión - LR



(c) Matriz de confusión - SVM

Figura 4.1: Predicciones de edificio

Para predecir el edificio, los tiempos de respuesta que tarda el sistema en devolver las predicciones de las 1111 muestras son de 11.5 s. para KNN, 1.5 s. para LR y 1.8 s. para SVM. Esto hace que el algoritmo de LR sea el que mejor desempeño logra, obteniendo un 100 % de acierto en las predicciones del edificio y tardando 1.5 s. en total para las 1111 muestras, lo que supone 1.35 ms el tiempo que tarda el sistema en predecir cada muestra.

4.2. Predicciones - Planta

Para predecir la planta primero se divide según los identificadores de edificio. Así se podrá tratar con coherencia cada problema de predicción teniendo en cuenta características propias de cada edificio y ese contexto específico.

4.2.1. Edificio 0:

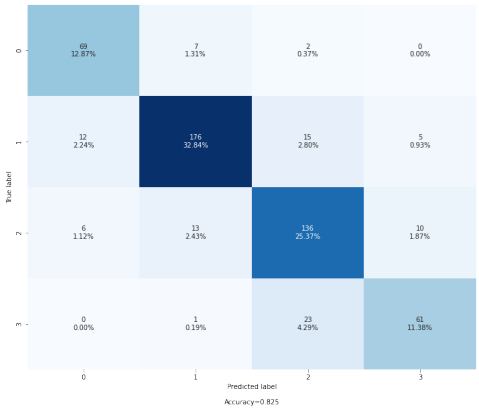
En el caso del edificio 0, no existen grandes diferencias entre la Regresión Logística Multiclase y SVM excepto por el tiempo de ejecución o entrenamiento, tal y como se muestra en el cuadro 4.2. Para SVM el tiempo es casi el doble que en los otros dos casos, pero también se consigue un modelo más pulido. El que obtiene peores resultados en este caso es KNN, se aleja bastante de la precisión obtenida con los otros dos algoritmos aunque mejora el tiempo de entrenamiento o ejecución con la predicción del edificio por el número de muestras, que también es menor. En este caso, LR y SVM ofrecen buenos resultados y son candidatos a ser modelos de producción para la predicción de la planta en el edificio 0, aunque seguramente se pueda mejorar la precisión con arquitecturas más complejas.

Cuadro 4.2: Predicciones de Planta - Edificio 0

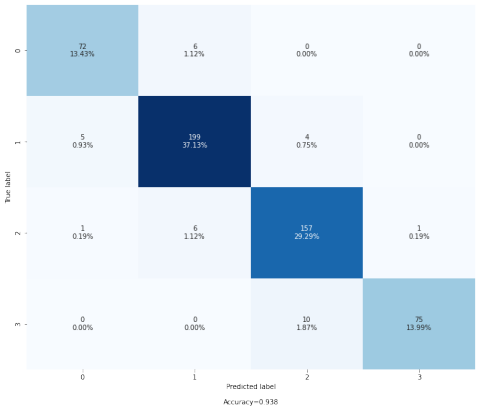
	KNN	LR	SVM
av_balanced_acc	98.17 %	98.9 %	98.93 %
variance	0.85	0.697	0.689
test_accuracy	81.82 %	92.84 %	93.44 %
t_train_ex	0.42 min	0.46 min	0.75 min
t_pred	1.98 ms	1.35 ms	1.71 ms

En la figura 4.2 se pueden diferenciar las matrices de confusión y como KNN parece que tiene cierta dificultad en la predicción de la planta 3, clasificándola erróneamente como si fuera planta 2. Aunque resulte evidente, también se puede identificar que la gran mayoría de clasificaciones erróneas siempre son entre plantas contiguas.

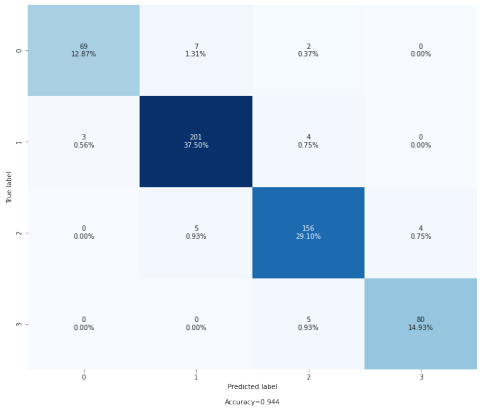
Para predecir la planta del edificio 0, los tiempos de respuesta que tarda el sistema en devolver las predicciones de las 1111 muestras son de 2.2 s. para KNN, 1.5 s. para LR y 1.9 s. para SVM. Esto hace que el algoritmo de SVM sea el que mejor desempeño logra, obteniendo un 93.44 % de acierto en las predicciones del edificio y tardando 1.9 s. en total para las 1111 muestras, lo que supone 1.71 ms el tiempo que tarda el sistema en predecir cada muestra.



(a) Matriz de confusión - KNN



(b) Matriz de confusión - LR



(c) Matriz de confusión - SVM

Figura 4.2: Predicciones de Planta - Edificio 0

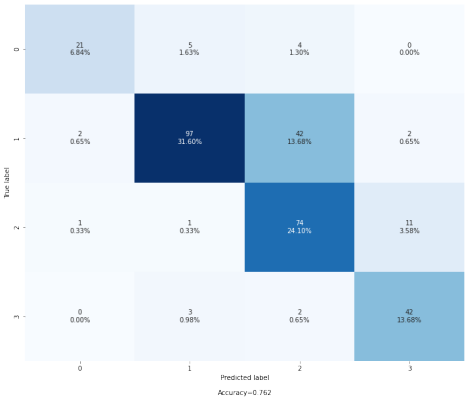
4.2.2. Edificio 1:

En el caso del edificio 1, no existen grandes diferencias entre la Regresión Logística Multiclase y SVM ni tan siquiera por el tiempo de ejecución o entrenamiento, tal y como se muestra en el cuadro 4.3. Este edificio presenta más dificultad que el anterior con los 3 algoritmos, aunque se obtiene una buena precisión media de entrenamiento, la varianza es mayor que con el edificio 0 y se obtiene una menor precisión de testeo, por lo que se trata de un modelo menos genérico. Para los tres algoritmos se tienen dificultades en la predicción de la planta 1, que confunden con la planta 0 y 2. Lo ideal en este caso sería identificar la disposición de estas plantas y ver como están representados los valores. Tal vez sería necesario definir un poco más los conjuntos

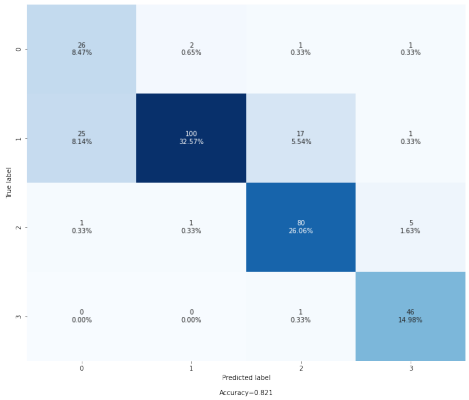
para afinar en la predicción. En la figura 4.3 se pueden diferenciar las matrices de confusión con los detalles expuestos anteriormente.

Cuadro 4.3: Predicciones de Planta - Edificio 1

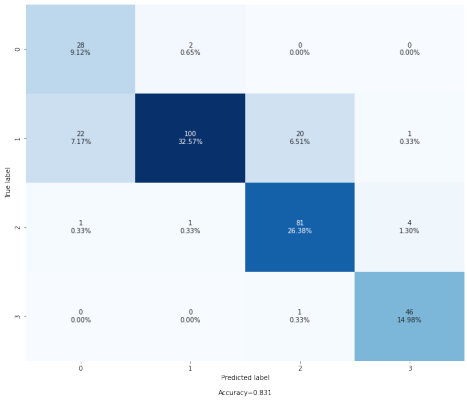
	KNN	LR	SVM
av_balanced_acc	98.12 %	97.87 %	97.67 %
variance	2.632	2.971	3.093
test_accuracy	78.06 %	86.61 %	88.56 %
t_train_ex	0.25 min	0.4 min	0.51 min
t_pred	1.62 ms	1.26 ms	1.35 ms



(a) Matriz de confusión - KNN



(b) Matriz de confusión - LR



(c) Matriz de confusión - SVM

Figura 4.3: Predicciones de Planta - Edificio 1

Para predecir la planta del edificio 1, los tiempos de respuesta que tarda el sistema en devolver las predicciones de las 1111 muestras son de 1.8 s. para KNN, 1.4 s. para LR y 1.5 s. para SVM. Esto hace que el algoritmo de SVM sea el que mejor desempeño logra, obteniendo un 88.56 % de acierto en las predicciones del edificio y tardando 1.5 s. en total para las 1111 muestras, lo que supone 1.35 ms el tiempo que tarda el sistema en predecir cada muestra.

4.2.3. Edificio 2:

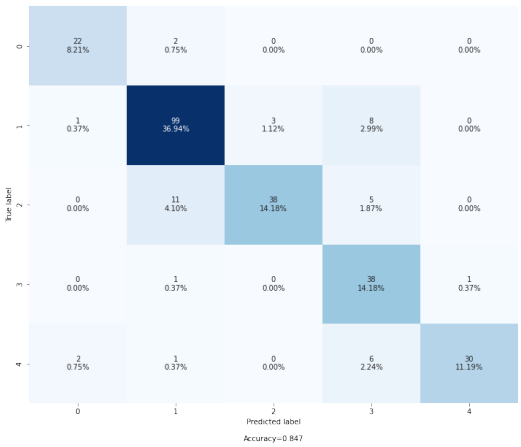
En el caso del edificio 2, no existen grandes diferencias entre la Regresión Logística Multiclase y SVM pasando del 90 % de precisión con datos de prueba excepto, de nuevo, por el tiempo de ejecución o entrenamiento que es del doble, tal y como se muestra en el cuadro 4.3. Para SVM el tiempo de entrenamiento es el doble que en la utilización de la Regresión Logística Multiclase, pero también se consigue un modelo más pulido y con menor varianza. El que obtiene peores resultados en este caso es KNN ya que se aleja de la precisión obtenida con los otros dos algoritmos aunque se trata del edificio que mejor predice KNN con un 84.63 % de precisión con datos de prueba. En este caso, LR y SVM ofrecen buenos resultados y son candidatos a ser modelos de producción para la predicción de la planta en el edificio 2, aunque seguramente se pueda mejorar la precisión con arquitecturas más complejas. En la figura 4.3 se pueden diferenciar las matrices de confusión con los detalles expuestos anteriormente.

Cuadro 4.4: Predicciones de Planta - Edificio 2

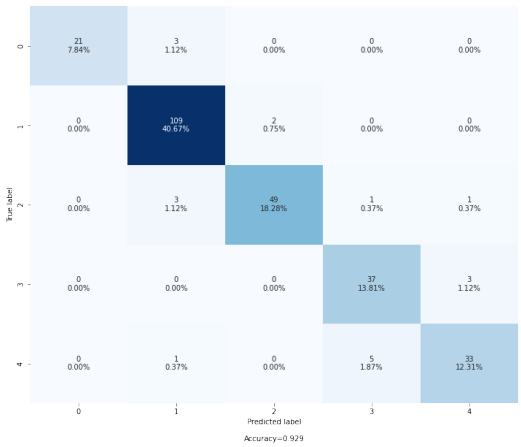
	KNN	LR	SVM
av_balanced_acc	97 %	98.61 %	98.9 %
variance	1.386	0.978	0.688
test_accuracy	84.63 %	90.71 %	91 %
t_train_ex	1.1 min	0.65 min	1.3 min
t_pred	2.16 ms	1.44 ms	1.44 ms

Para predecir la planta del edificio 2, los tiempos de respuesta que tarda el sistema en devolver las predicciones de las 1111 muestras son de 2.4 s. para KNN, 1.6 s. para LR y 1.6 s. para SVM. Esto hace que el algoritmo de SVM sea el que mejor desempeño logra, obteniendo un 91 % de acierto en las predicciones del edificio y tardando 1.6 s. en total para las 1111 muestras, lo que supone 1.44 ms el tiempo que tarda el sistema en predecir cada muestra.

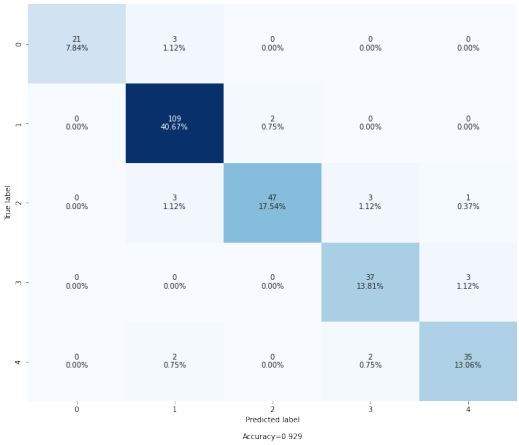
Respecto a otros artículos, cabe mencionar que los resultados vuelven a ser superiores respecto a los valores SVM en [47] para la predicción de la planta, pero ligeramente inferiores a los resultados con RandomForest.



(a) Matriz de confusión - KNN



(b) Matriz de confusión - LR



(c) Matriz de confusión - SVM

Figura 4.4: Predicciones de Planta - Edificio 2

4.3. Arquitectura de modelos - Clustering

En primer lugar, se decide la utilización de DBSCAN para la parte de Clustering inpirado en el artículo [55]. El primer motivo fue por intentar agrupar por densidades los distintos grupos y se prefería no especificar el número de clústers, se quería probar a que automáticamente el algoritmo, según la densidad de puntos en las distintas localizaciones agrupara por subconjuntos. DBSCAN puede encontrar clusters con formas geométricas arbitrarias, por lo que no habría problema entendiendo estos subconjuntos con distintos tipos de formas geométricas. Además,

DBSCAN es robusto detectando outliers. El problema es que DBSCAN no es enteramente determinista ni puede agrupar conjuntos de datos correctamente con grandes diferencias en las densidades [17]. Lo que pasó con estos análisis es que realmente no encontraba los subgrupos de manera coherente porque no existen claramente distintos focos densos de puntos y los puntos borde que marcarían los bordes de las habitaciones son alcanzables desde más de un clúster y pueden ser etiquetados con cualquiera de estos clusters al tratarse de escenarios heterogéneos. Como posteriormente confirmo al revisar el artículo [39], debido a la heterogeneidad de este conjunto de datos y a su distribución no homogénea, algunas muestras quedan excluídas o marcadas como ruido. En otro de los experimentos con uno de los edificios se encuentra que incluso se representa un único clúster y toma el resto de puntos válidos como ruido. Este no es el resultado que se espera encontrar y por eso el interés del anterior artículo en su mejora.

Finalmente, se decide la utilización de K-Means ya que es el método que mejores resultados ofrece con este tipo de distribuciones de datos. Posteriormente se pasará a evaluar el desempeño del algoritmo en las arquitecturas construidas para conseguir la predicción de la localización.

4.4. Predicciones - Localización

En estas predicciones de localización se dividen los experimentos según arquitectura de modelos K-Means-KNN, K-Means-LR y K-Means-SVM. Cabe recordar que esta variable de "localización" se trata de una variable nueva creada en el proceso de limpieza de los datos (ver punto 4 del apartado 3.4) con el objetivo de establecer un único valor numérico por cada valor de altura, latitud y longitud. Es decir, una única variable que representaría un único espacio que ha sido registrado previamente en la fase de entrenamiento y que el algoritmo recuerda. En cada uno de los experimentos se realiza un análisis por identificador de edificio, planta y valor de K con el objetivo de predecir ese valor de localización único registrado previamente por los datos de entrenamiento. Por cada conjunto o experimento la duración ronda entre los 6 y los 8 segundos. Estos experimentos nos proporcionan los valores de K y los algoritmos que mejor funcionan para cada edificio y planta, por lo que en producción se podrá establecer qué arquitectura es la ganadora en base al edificio y planta.

A continuación, se muestran los resultados con la arquitectura K-Means-KNN que es la que obtiene los peores resultados de las tres arquitecturas propuestas. El Edificio 0 es el que peores resultados obtiene con esta arquitectura y resulta curioso que para el Edificio 1, el valor de K ganador sería el mismo (K=10). Otro resultado curioso es que las precisiones en el Edificio 1 son ligeramente superiores a las obtenidas con SVM para el Edificio 1.

4.4.1. KNN - Edificio 0:

Cuadro 4.5: Predicciones de Localización - KNN - Edificio 0

	K2	K4	K6	K8	K10	K12
f0	67.8 %	73.8 %	67.7 %	72.1 %	76.4 %	73.7 %
f1	65.5 %	69.5 %	75.9 %	77.7 %	72.4 %	76.1 %
f2	58.3 %	62.1 %	73.8 %	70.4 %	69.2 %	72.3 %
f3	59.9 %	65.4 %	65.6 %	69 %	63.4 %	65.8 %

4.4.2. KNN - Edificio 1:

Cuadro 4.6: Predicciones de Localización - KNN - Edificio 1

	K2	K4	K6	K8	K10	K12
f0	80.3 %	83.2 %	85.2 %	85.2 %	84.7 %	83.9 %
f1	75.7 %	72.3 %	72.9 %	76.1 %	78.9 %	78 %
f2	80.7 %	85 %	84.9 %	84.8 %	87.6 %	86.8 %
f3	70.5 %	81.7 %	80.5 %	80.6 %	84.6 %	82.8 %

4.4.3. KNN - Edificio 2:

Cuadro 4.7: Predicciones de Localización - KNN - Edificio 2

	K2	K4	K6	K8	K10	K12
f0	84.8 %	85.4 %	86.8 %	85.1 %	84.9 %	83.9 %
f1	81.3 %	75.9 %	76.3 %	77.5 %	78.3 %	72.9 %
f2	69.4 %	72 %	73.5 %	72.5 %	75 %	79.6 %
f3	74.6 %	74.7 %	76.1 %	72.3 %	74.7 %	73.3 %
f4	71.9 %	66.7 %	65.1 %	65.4 %	62.4 %	62.6 %

A continuación, se muestran los resultados con la arquitectura K-Means-LR que es la que obtiene los mejores resultados de las tres arquitecturas propuestas, aunque también obtiene valores nulos para según que planta, edificio y valor de K. Esto tiene sentido al encontrar el valor de K correcto que hace que los subconjuntos sea separables. En este caso, y como se puede ver en las siguientes tablas, se obtienen valores altos de precisión que podrían ser mejorados si aplicáramos algunos métodos concretos a esta arquitectura que hiciera los grupos perfectamente separables.

4.4.4. LR - Edificio 0:

Cuadro 4.8: Predicciones de Localización - LR - Edificio 0

	K2	K4	K6	K8	K10	K12
f0	NaN	91.7 %	NaN	91 %	91 %	91 %
f1	NaN	NaN	89 %	92.8 %	95.2 %	92.8 %
f2	NaN	NaN	NaN	85.2 %	93.2 %	77.5 %
f3	NaN	NaN	NaN	68.3 %	NaN	78.3 %

4.4.5. LR - Edificio 1:

Cuadro 4.9: Predicciones de Localización - LR - Edificio 1

	K2	K4	K6	K8	K10	K12
f0	NaN	85.8 %	91.4 %	91.4 %	93.4 %	91.4 %
f1	82.5 %	NaN	87.5 %	87.5 %	91.1 %	91.1 %
f2	NaN	NaN	92.5 %	83.6 %	97 %	89.5 %
f3	NaN	85 %	84.8 %	85 %	84.4 %	83.5 %

4.4.6. LR - Edificio 2:

Cuadro 4.10: Predicciones de Localización - LR - Edificio 2

	K2	K4	K6	K8	K10	K12
f0	96.7 %	90 %	100 %	94.2 %	100 %	96.7 %
f1	91.7 %	91.7 %	90 %	90 %	90 %	79.2 %
f2	NaN	NaN	NaN	88.9 %	90 %	94.3 %
f3	NaN	NaN	90 %	89.4 %	100 %	89.1 %
f4	92.1	91.5	48.8 %	52.2 %	48.8 %	47.2 %

Por último, se muestran los resultados con la arquitectura K-Means-SVM que es la que obtiene los resultados más fiables y constantes de las tres arquitecturas propuestas independientemente de los valores de indentificador de edificio, planta y K. En este caso, y como se puede ver en las siguientes tablas, se obtienen valores de precisión mejorables para según que subconjunto si aplicáramos algunos métodos concretos a esta arquitectura que hiciera que los subconjuntos estuvieran mucho mejor definidos.

4.4.7. SVM - Edificio 0:

Cuadro 4.11: Predicciones de Localización - SVM - Edificio 0

	K2	K4	K6	K8	K10	K12
f0	74.11 %	74.11 %	74.11 %	74.11 %	82.82 %	78.22 %
f1	74.70 %	74.70 %	74.70 %	74.70 %	80.70 %	82.23 %
f2	69.32 %	69.32 %	69.32 %	69.32 %	72.82 %	73.41 %
f3	67.24 %	67.24 %	67.24 %	67.24 %	72.20 %	72.96 %

4.4.8. SVM - Edificio 1:

Cuadro 4.12: Predicciones de Localización - SVM - Edificio 1

	K2	K4	K6	K8	K10	K12
f0	81.19 %	81.19 %	81.19 %	81.19 %	84.69 %	83.43 %
f1	77.88 %	77.88 %	77.88 %	77.88 %	75.48 %	73.27 %
f2	89.17 %	89.17 %	89.17 %	89.17 %	91.51 %	90.43 %
f3	77.54 %	77.54 %	77.54 %	77.54 %	82.46 %	83.26 %

4.4.9. SVM - Edificio 2:

Cuadro 4.13: Predicciones de Localización - SVM - Edificio 2

	K2	K4	K6	K8	K10	K12
f0	87.88 %	87.88 %	87.88 %	87.88 %	88.03 %	86.79 %
f1	85.95 %	85.95 %	85.95 %	85.95 %	85.35 %	79.22 %
f2	84.46 %	84.46 %	84.46 %	84.46 %	86.84 %	88.51 %
f3	83.37 %	83.37 %	83.37 %	83.37 %	83.26 %	81.06 %
f4	74.08 %	74.08 %	74.08 %	74.08 %	64.60 %	64.68 %

4.5. Infraestructura y predicciones

MLFlow permanece escuchando desde el servidor una vez ejecutamos el siguiente comando desde una terminal a través de SSH en la instancia de Google Cloud:

```
mlflow server --backend-store-uri
        postgresql://mlflow:3210_key_9880@localhost/mlflow
        --default-artifact-root gs://example_bucket_data_science_1
        -h 0.0.0.0 -p 8080
```

Con el comando anterior se establece la base de datos a la que se quiere apuntar, la ruta donde se almacenarán los artefactos y se define que se ejecute en la instancia en el puerto 8080. Para más detalle de configuración consultar la guía en [2]. Para más detalle en cuanto al registro y obtención de un modelo desde la librería de modelos de MLFlow se puede consultar el siguiente enlace [32].

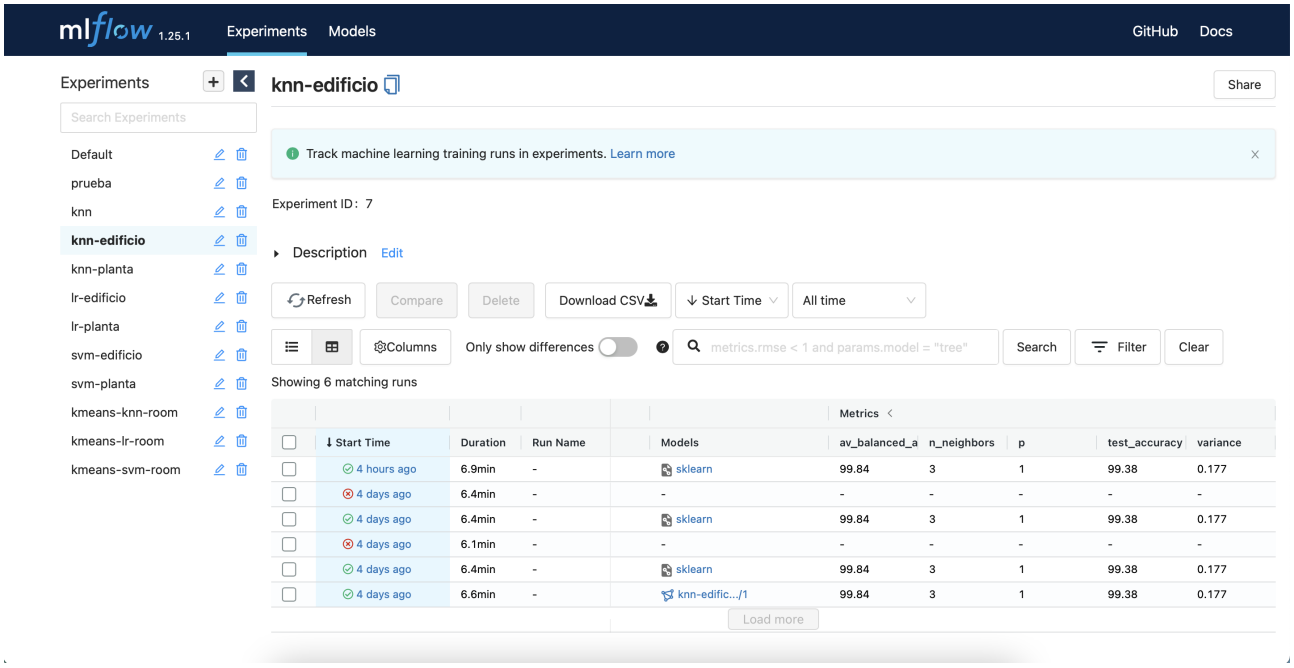


Figura 4.5: Pantalla principal de experimentos - MLFlow

Para acceder al tablero de MLFlow tan sólo tendríamos que acceder en un navegador a la dirección: `http://34.88.54.190:8080`. En la pantalla general se encuentra el registro de experimentos por nombre, desde ahí se puede acceder a cada uno de los experimentos. Para cada uno de ellos se muestran las ejecuciones asociadas con información como la duración de la ejecución, el nombre, el usuario, la versión, el modelo y las métricas registradas. Estos datos pueden ser filtrados en modo de tabla y se pueden descargar los resultados en un csv (ver Figura 4.5). En la vista general de modelos se pueden comprobar los distintos modelos, sus versiones y cuando fueron modificados por última vez (ver Figura H.1). Si pulsamos en alguno de los modelos o en algunas de sus versiones aparecerá la siguiente pantalla en la que entra en detalle de cada modelo, su descripcion, sus etiquetas y más detalle sobre sus versiones en producción

o en test (ver Figura H.2).

Si pulsamos sobre alguna de sus versiones se obtiene información de la versión en detalle, una descripción, los parámetros asociados, las métricas, etiquetas y artefactos. Estos artefactos son los que han sido registrados previamente cuando se ha ejecutado el componente. Podemos guardar cualquier tipo de artefacto. Este artefacto será accesible por la ruta de la versión en concreto (ver Figura H.3). Además de los artefactos que se guardan en la ruta específica de la versión se pueden obtener las predicciones sobre el modelo de la versión con ayuda de la guía y apuntando al string que representa la versión en cada caso. También se pueden acceder a los artefactos registrados, como por ejemplo, imágenes con las matrices de confusión del modelo (ver Figura H.4).

Al contar con una instancia de Google Cloud, se va a poder acceder a cualquier material o artefactos que han sido cargados en cada una de las ejecuciones desde cualquier lugar. Además se van a poder predecir valores acorde a cada modelo apuntando a las rutas de las versiones. Esto ofrece la posibilidad de tener modelos en producción y apuntar cada una de las aplicaciones a cada una de las versiones del modelo que sean apropiadas.

Finalmente se podrá realizar una prueba de obtención de predicciones apuntando a un modelo registrado en MLFlow. En este caso se podrá hacer uso del siguiente template:

utils/prediccion.ipynb.

En la primera celda y en el caso de ejemplo del template, se requieren 1111 predicciones apuntando al modelo "knn-edificio" para obtener las predicciones en base al dataset de test. Se utilizarán posteriormente las potencias de los registros de prueba para comprobar si tiene sentido lo que devuelve y el tiempo estimado de duración que tarda en obtener las predicciones. En este caso, la predicción tarda 12.7 segundos en obtener las 1111 predicciones, calcular la métrica de precisión del modelo y obtener el valor 99.37 %. Se debe tener en cuenta que en este primer caso ya se tienen los ficheros con los valores limpios para poder requerir la predicción.

En la segunda celda se puede encontrar un modelo para conseguir realizar el proceso de limpieza y así poder requerir correctamente las predicciones partiendo desde el punto inicial en el que se cuenta únicamente con un fichero .csv y con los datos sin pasar por el pre-procesado. Otra de las mejoras sería poder automatizar este proceso para que se puedan pasar a predecir tambien valores que no han sido pre-procesados.

Capítulo 5

Trabajo futuro, debate y conclusiones

5.1. Adaptación a dataset Bluetooth Low Energy

El sistema tiene una fácil adaptación a trabajar con registros BLE al tratarse de la misma unidad que es la potencia registrada por los puntos de acceso. Este sería el primer requerimiento, si se tienen unos datos de entrada con unas potencias asociadas registradas por cada punto de acceso respecto a cada dispositivo en filas, tan sólo será cuestión de evaluar el dataset y realizar una limpieza de datos apropiada. Este componente de limpieza de datos por tanto, tendría dos subcomponentes que trabajarían cada uno de ellos de forma independiente según se trate de registros BLE o Wi-Fi.

Una vez estos datos han pasado la fase de limpieza de datos habría que adaptar el resto de componentes conforme al número de puntos de acceso, pero no hay una labor mucho más importante en esto. Las funciones de apoyo podrían identificar que se trata de registros BLE y así, aplicar unas transformaciones concretas. La infraestructura y la base estarían creadas, con la ventaja de que se va a poder realizar un seguimiento por cada ejecución que se realice y se podrá iterar en la adaptación del sistema a los registros BLE.

Esto podría otorgar funcionalidad extra al sistema, como por ejemplo el de verificación de los resultados por medio de ambas técnicas. Si tenemos varios modelos entrenados con datos Wi-Fi y BLE, se podría construir una arquitectura en la que ambos sistemas pudieran aportar y ayudar en la calibración y validación de ciertos registros en contextos complicados o con malos resultados con una de las técnicas anteriores en entornos heterogéneos.

5.2. Transfer Learning e Incremental Learning - IoT

El sistema obtiene los datos desde una fuente fija de archivos en una carpeta en local. En primer lugar, hay tener en cuenta que esta fuente puede cambiar, tan sólo se trataría de añadir los parámetros necesarios de entrada al componente en el fichero YAML. Estos pueden ser en cualquier formato: URL endpoint, ruta de archivo en la nube o por ejemplo la ruta del flujo de datos. Esta fuente de datos podría ser un flujo de datos contínuo, batches o datos en tiempo real de un sistema IoT. A través de esta fuente de datos se cargaría un objeto Dataframe. Estas muestras que se reciben pueden ser almacenadas en una base de datos intermedia y ser consumida a través de una API, pero existe la alternativa de que se puedan registrar los registros de entrenamiento como artefactos en la misma ejecución de los modelos y esos datos puedan ser cargados nuevamente para mezclar con los nuevos y volver a entrenar el modelo gracias al uso de la API de MLflow.

Para entrenar el modelo se podría hacer uso de spark_udf. MLFlow ofrece posibilidades de integración con PySpark para seguimiento de los modelos. En este caso se podría ver la evolución del modelo haciendo uso de las funcionalidades de Spark tras pasar por un objeto Pipeline y lo interesante sería descubrir qué patrones distintos o cambios de características en los datos puede dar lugar a otro resultado en el modelo entrenado, con la computación distribuida a escala que otorga Spark en cuanto al manejo de un gran volúmen de datos y del uso del paradigma MapReduce. En este punto se podrían incorporar arquitecturas Transfer Learning, Incremental Learning o Deep Learning que puedan hacer uso de estas funcionalidades al integrar herramientas como Spark Streaming o Structured Streaming. De la misma manera que anteriormente, y gracias a que MLFlow fue creado por Databricks, las integraciones con estas herramientas proporcionan un entorno que permite llevar estas herramientas a producción muy facilmente.

5.3. Monitorización y alertas

Great Expectations (github.com/great-expectations) sería una herramienta a integrar que proporcionaría al sistema análisis en cuanto a la calidad de los datos. Este framework ayuda a los equipos de datos a registrar pruebas de datos a través de las denominadas “expectativas”. Se podrían registrar pruebas declarativas como por ejemplo:

- Se espera que esta tabla tenga entre X e Y número de filas.
- Se espera que los valores nulos no superen el 20 % del total.

- Se espera que el formato de la columna de fecha sea MM-DD-YYYY.

Otra de las ventajas es que ofrece una plataforma en la que renderizar las pruebas a modo formulario y donde se pueden generar reportes. Además tiene integración con herramientas como Pandas, Jupyter Notebooks, Spark o Databricks, por lo que podría encajar en el puzzle de tecnologías del sistema y aportaría información valiosa, sobre todo si se empieza a utilizar este sistema en producción para predicciones en tiempo real. Se podrán establecer alertas ante posibles valores incorrectos que lleguen a través del flujo de datos.

5.4. Conclusiones

Este proyecto finalmente ofrece una solución "end-to-end" para que, a través del pipeline desarrollado y puesto en producción de manera flexible siguiendo el enfoque MLOps y sus buenas prácticas para evitar sobre-coste de mantenimiento en el futuro [46], se puedan obtener distintas predicciones como el identificador de edificio, la planta o la localización única. En concreto, para este sistema se consiguen muy buenas medidas de precisión en la predicción del identificador del edificio y de la planta. También se obtienen medidas razonablemente buenas en la predicción de la localización en base a los subgrupos creados en la etapa de Clustering y finalizando la arquitectura con alguno de los 3 modelos propuestos.

Para la realización del proyecto se ha utilizado la cuenta gratuita proporcionada por Google Cloud con 300€ de crédito y 90 días de uso. No se ha llegado a utilizar todo el crédito, ya que con la instacia creada tan sólo se ha llegado a gastar 0.25€ por día (en ejecución). El sistema por tanto, es barato, fácil de construir, escalable, flexible y haciendo uso de herramientas open-source.

En cualquier momento se puede actualizar cualquier componente y Ploomber automáticamente ejecutará el pipeline desde donde identifica el componente actualizado o nuevo. Se obtendrá finalmente como salida (output) de Ploomber en la terminal el siguiente registro de las etapas tras la ejecución del pipeline:

name	Ran?	Elapsed (s)	Percentage
KNN Planta	True	114.394	15.1043
KNN Room	True	642.965	84.8957
Get data & evaluate	False	0	0
Report data	False	0	0

Lo ideal sería poder utilizar esta infraestructura para realizar predicciones en tiempo real utilizando datos de un sistema IoT y que fueran consumidos por una aplicación. La funcionalidad podrá ser permitida ya que se van a poder registrar y cargar los datos y las predicciones desde cualquier dispositivo con conexión a internet. También se podrían cargar todos los datasets de una carpeta y se podría realizar el análisis en paralelo o por lotes, dependiendo por ejemplo del clúster. Otras de las mejoras podría ser la interfaz del template de predicciones, se podría también contar con un endpoint que pudiera cargar los datos ya limpios y preparados para el modelo, o incorporar esa etapa en el mismo pipeline en producción para que sea ajeno al usuario.

Por lo que las líneas futuras van por la adaptación de los componentes para trabajar con datos en tiempo real además de trabajar con datos fijos y estáticos. Se puede empezar a adaptar un componente en paralelo que fuera limpiando los datos que van llegando de BLE, para que así posteriormente pueda tambien ser adaptado a Wi-Fi. Se propone el estudio de estos cambios con Spark y con Structured Streaming para ver que ventajas e inconvenientes presenta cada uno. Las herramientas facilitadas en el sistema ofrecen integración con cualquier otro componente para realizar análisis de grandes volúmenes de datos y en tiempo real. Finalmente, se podrían llegar a implementar arquitecturas más complejas para que así las predicciones ofrezcan buenos resultados en cualquier contexto y puedan ser adaptadas a cualquier entorno con mucha facilidad.

Apéndice A

Infograma de contextualización

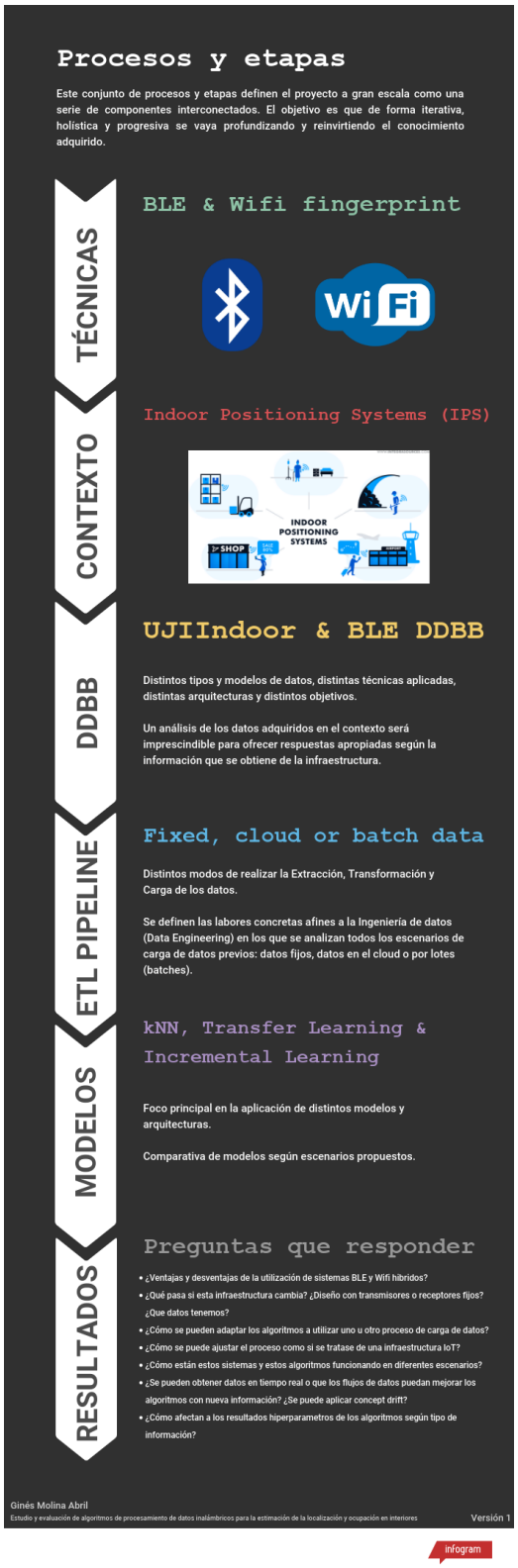


Figura A.1: Diagrama y contextualización de las diferentes etapas de la propuesta.

Apéndice B

Cronograma de tareas

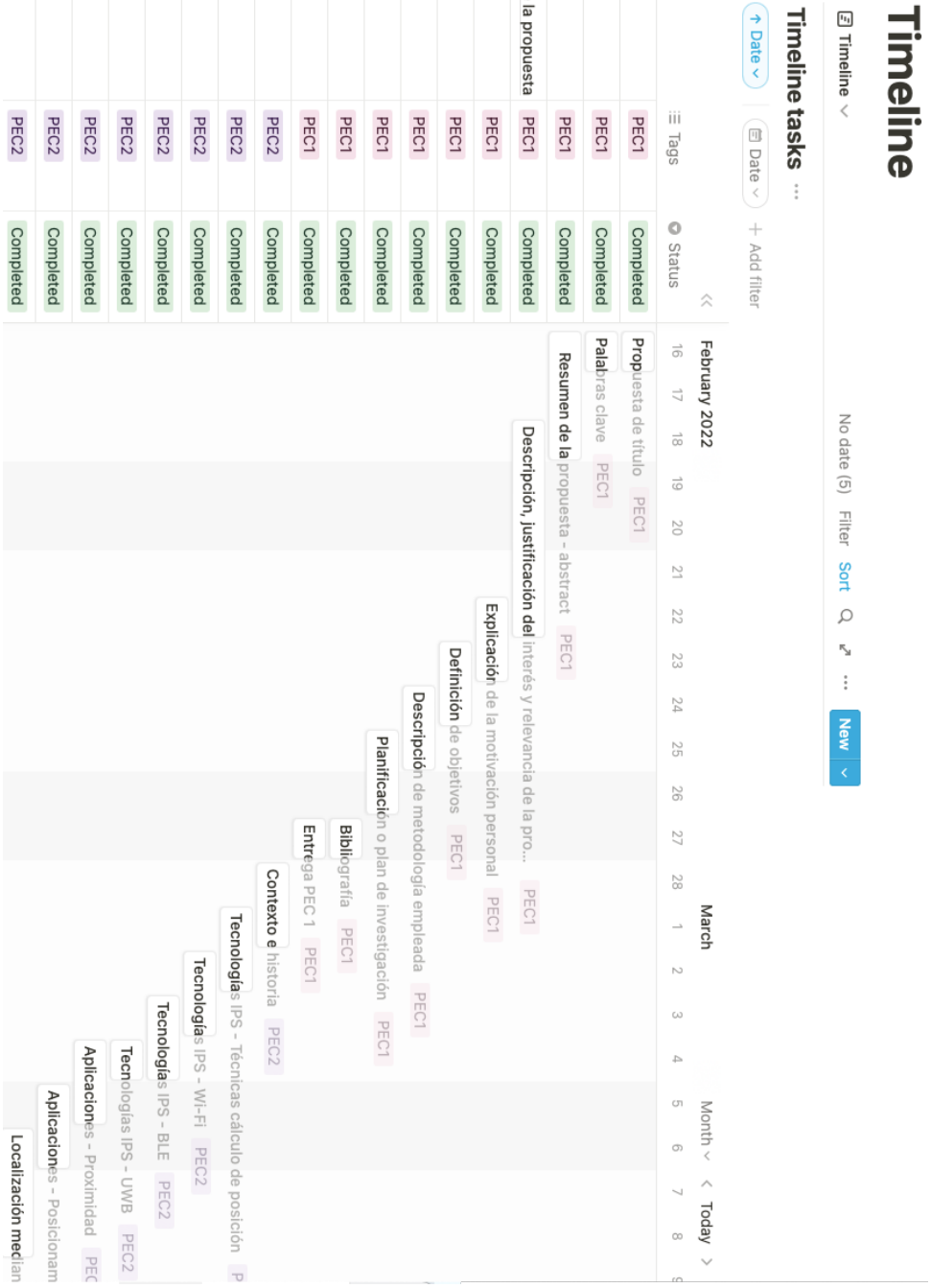
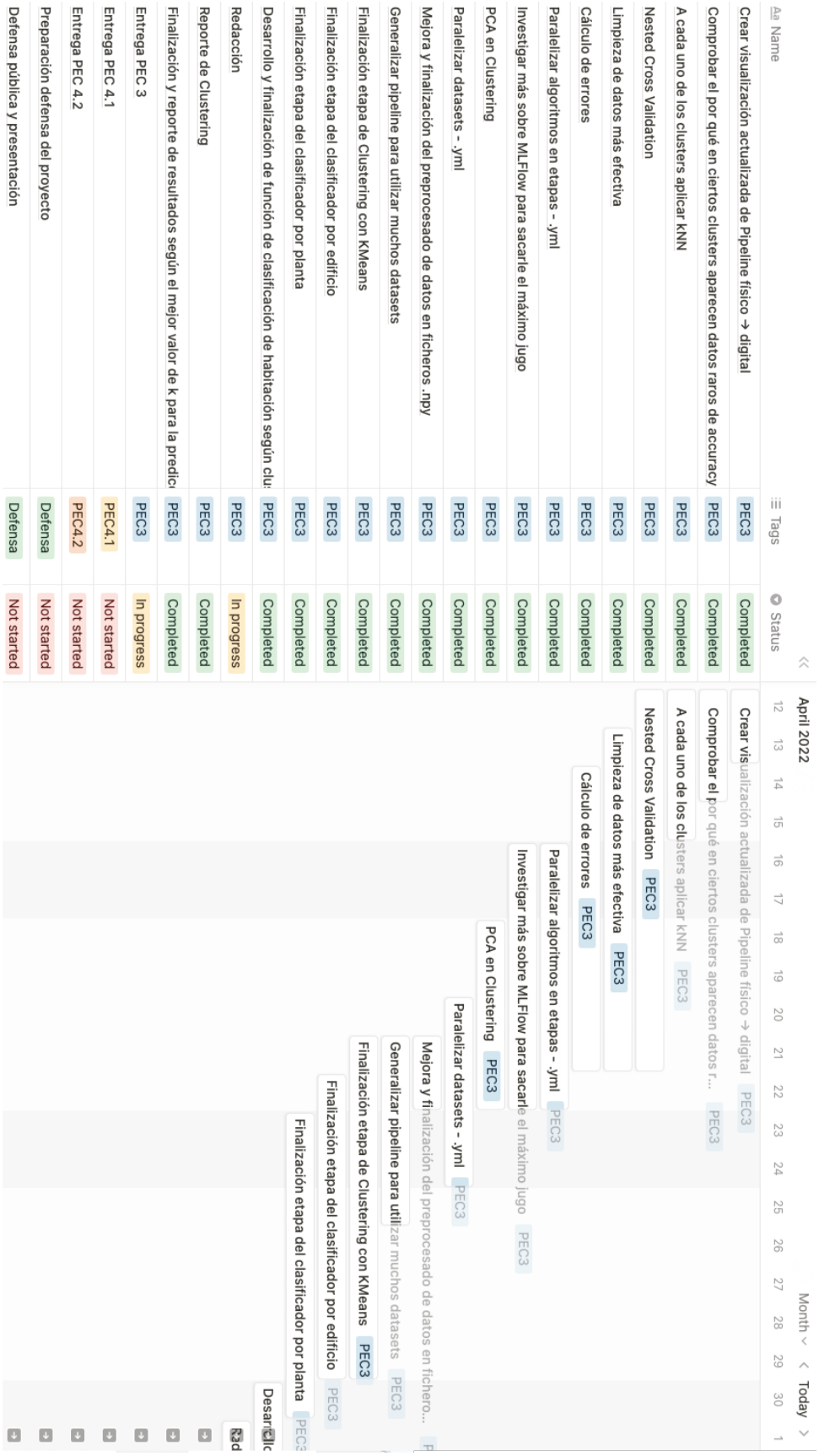


Figura B.1: Workflow - Sección 1

Name	Tags	Status
Localización mediante BLE	PEC2	Completed
Modelado - Algoritmo k-NN y SVM	PEC2	Completed
Modelado - Clustering	PEC2	Completed
Modelado - Incremental Learning	PEC2	Completed
Modelado - Transfer Learning	PEC2	Completed
Modelado - Ensemble models	PEC2	Completed
Bibliografía	PEC2	Completed
Entrega PEC2	PEC2	Completed
Research sobre infraestructura	PEC3	Completed
Puesta en marcha del repositorio GIT	PEC3	Completed
Puesta en marcha del entorno con anaconda	PEC3	Completed
Primer prototipo de Jupyter notebooks	PEC3	Completed
Puesta en marcha del framework Ploomber	PEC3	Completed
Puesta en marcha de MLFlow	PEC3	Completed
Primer prototipo de la etapa de reporte	PEC3	Completed
Primer prototipo de la etapa de limpieza de datos	PEC3	Completed
Primer prototipo de etapa de entrenamiento y clustering	PEC3	Completed
Primer prototipo de visualización	PEC3	Completed
Mejora del diagrama de etapas	PEC3	Completed
Implementación sistema Logging	PEC3	Completed
Separar dataframe por clusters	PEC3	Completed
Group by z - planta o altura previo a Clustering	PEC3	Completed
Entrenar modelo Clustering y realizar predicciones	PEC3	Completed

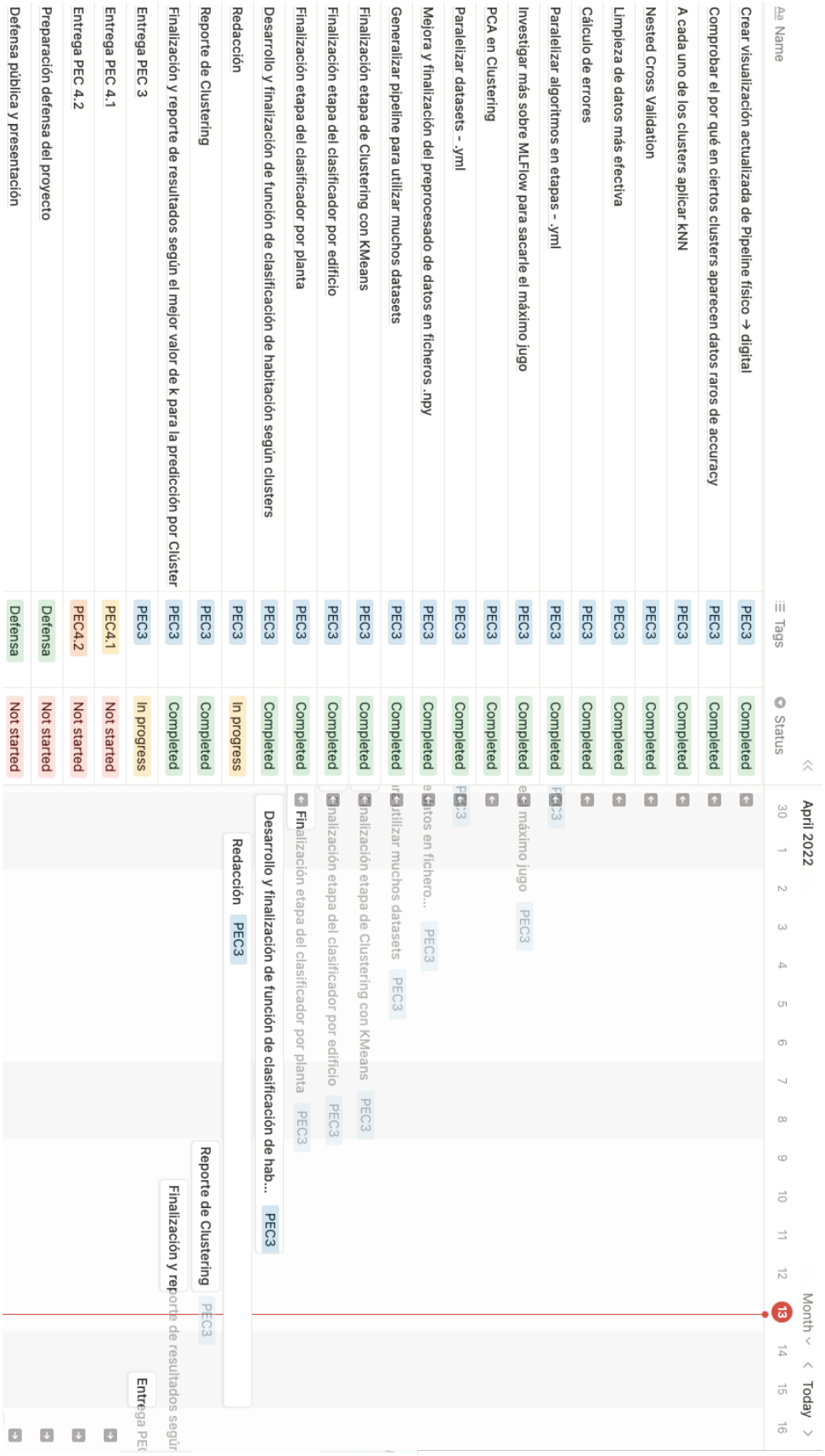
Figura B.2: Workflow - Sección 2

Figura B.3: Workflow - Sección 3



Name	Tags	Status
Crear visualización actualizada de Pipeline físico → digital	PEC3	Completed
Comprobar el por qué en ciertos clusters aparecen datos raros de accuracy	PEC3	Completed
A cada uno de los clusters aplicar kNN	PEC3	Completed
Nested Cross Validation	PEC3	Completed
Limpieza de datos más efectiva	PEC3	Completed
Cálculo de errores	PEC3	Completed
Paralelizar algoritmos en etapas - .yml	PEC3	Completed
Investigar más sobre MLFlow para sacarle el máximo jugo	PEC3	Completed
PCA en Clustering	PEC3	Completed
Paralelizar datasets - .yml	PEC3	Completed
Mejora y finalización del preprocesado de datos en ficheros .npy	PEC3	Completed
Generalizar pipeline para utilizar muchos datasets	PEC3	Completed
Finalización etapa de Clustering con KMeans	PEC3	Completed
Finalización etapa del clasificador por edificio	PEC3	Completed
Finalización etapa del clasificador por planta	PEC3	Completed
Desarrollo y finalización de función de clasificación de habitación según clu	PEC3	Completed
Redacción	PEC3	In progress
Reporte de Clustering	PEC3	Completed
Finalización y reporte de resultados según el mejor valor de k para la predic	PEC3	Completed
Entrega PEC 3	PEC3	In progress
Entrega PEC 4.1	PEC4.1	Not started
Entrega PEC 4.2	PEC4.2	Not started
Preparación defensa del proyecto	Defensa	Not started
Defensa pública y presentación	Defensa	Not started

Figura B.4: Workflow - Sección 4



Name	Tags	Status
Crear visualización actualizada de Pipeline físico → digital	PEC3	Completed
Comprobar el por qué en ciertos clusters aparecen datos raros de accuracy	PEC3	Completed
A cada uno de los clusters aplicar kNN	PEC3	Completed
Nested Cross Validation	PEC3	Completed
Limpieza de datos más efectiva	PEC3	Completed
Cálculo de errores	PEC3	Completed
Paralelizar algoritmos en etapas - .yml	PEC3	Completed
Investigar más sobre MLFlow para sacarle el máximo jugo	PEC3	Completed
PCA en Clustering	PEC3	Completed
Paralelizar datasets - .yml	PEC3	Completed
Mejora y finalización del preprocesado de datos en ficheros .npy	PEC3	Completed
Generalizar pipeline para utilizar muchos datasets	PEC3	Completed
Finalización etapa de Clustering con KMeans	PEC3	Completed
Finalización etapa del clasificador por edificio	PEC3	Completed
Finalización etapa del clasificador por planta	PEC3	Completed
Desarrollo y finalización de función de clasificación de habitación según clusters	PEC3	Completed
Redacción	PEC3	In progress
Reporte de Clustering	PEC3	Completed
Finalización y reporte de resultados según el mejor valor de k para la predicción por Clúster	PEC3	Completed
Entrega PEC 3	PEC3	In progress
Entrega PEC 4.1	PEC4.1	Not started
Entrega PEC 4.2	PEC4.2	Not started
Preparación defensa del proyecto	Defensa	Not started
Defensa pública y presentación	Defensa	Not started

Figura B.5: Workflow - Sección 5

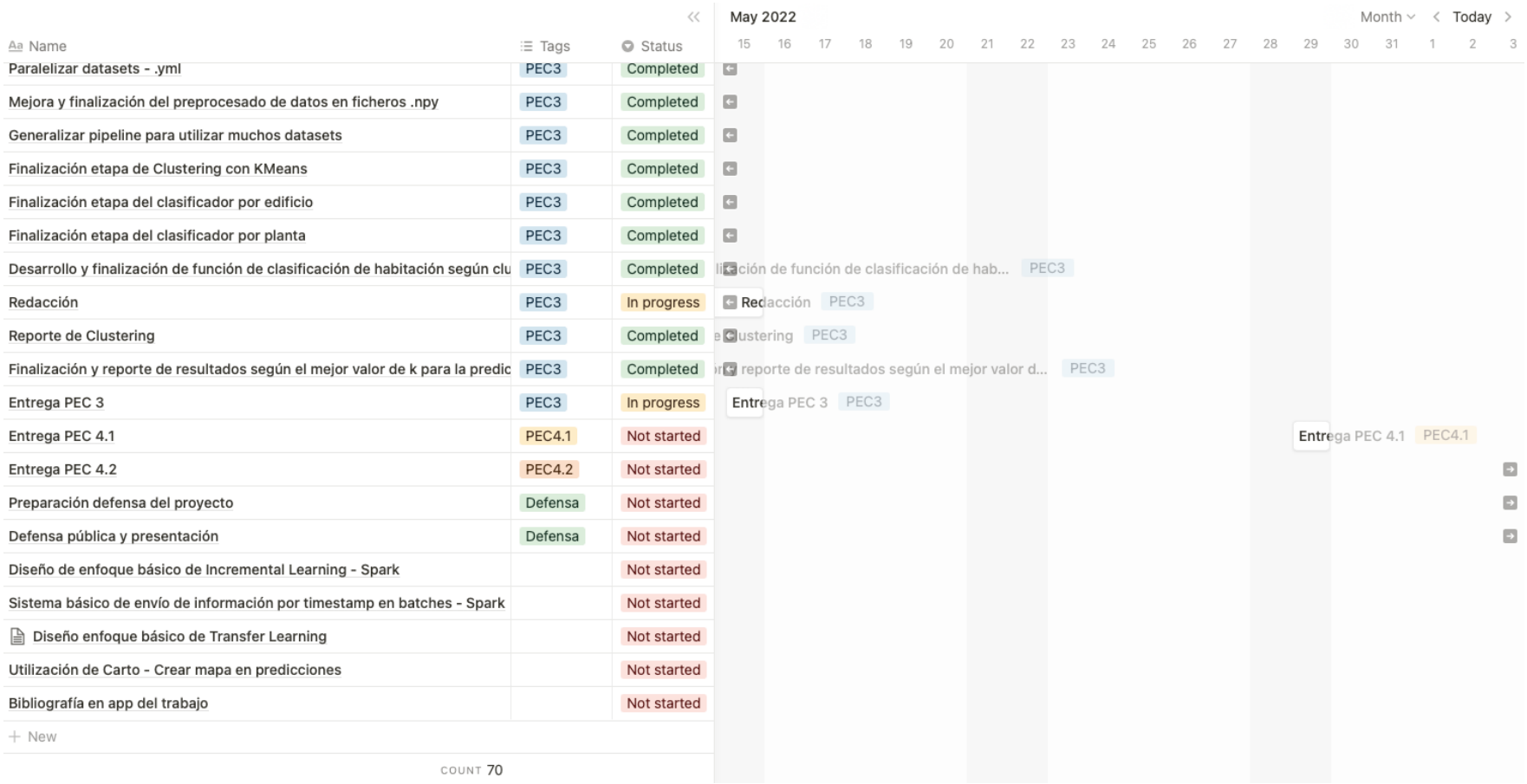
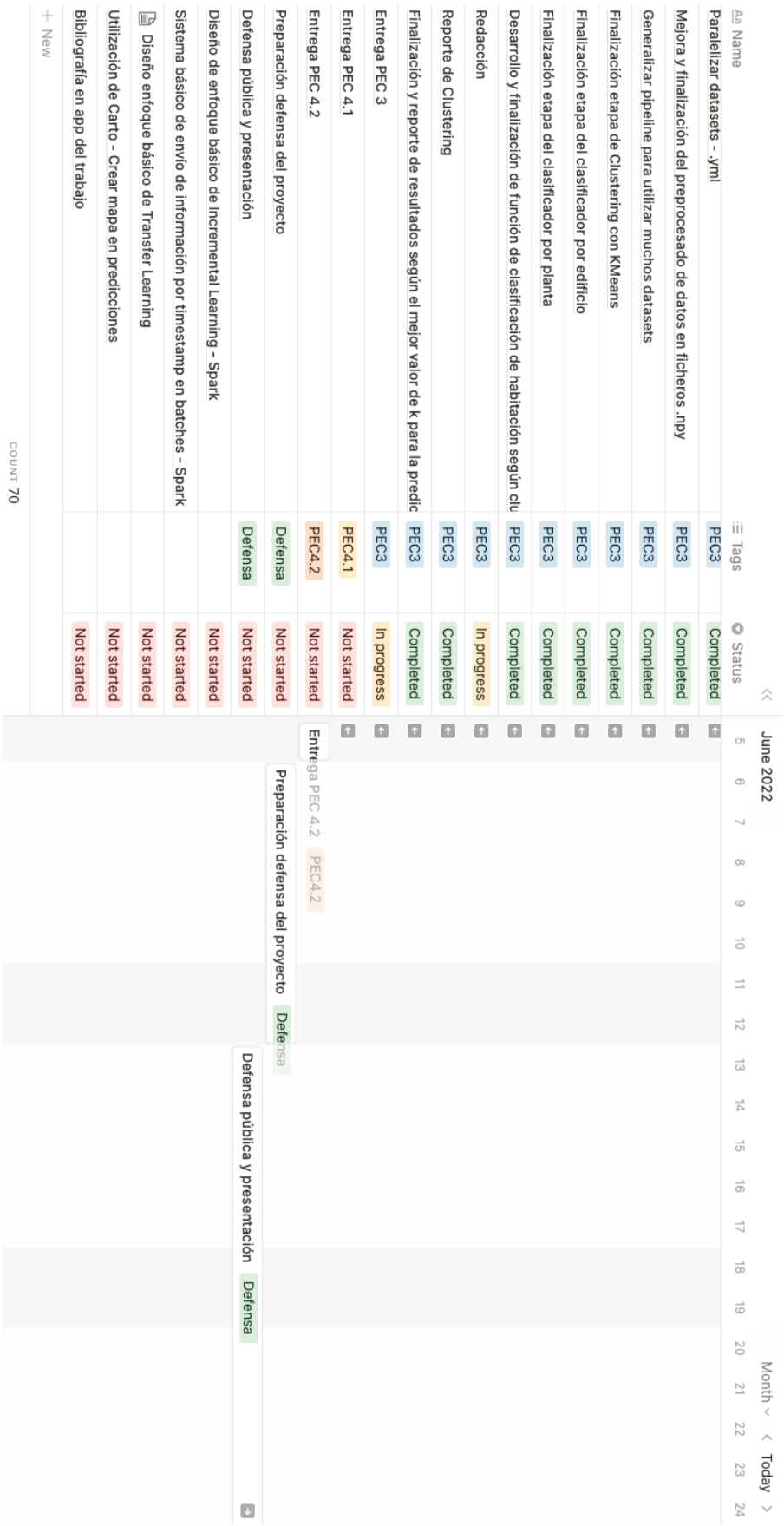


Figura B.6: Workflow - Sección 6



Name	Tags	Status
Paralelizar datasets - .yml	PEC3	Completed
Mejora y finalización del preprocesado de datos en ficheros .npy	PEC3	Completed
Generalizar pipeline para utilizar muchos datasets	PEC3	Completed
Finalización etapa de Clustering con KMeans	PEC3	Completed
Finalización etapa del clasificador por edificio	PEC3	Completed
Finalización etapa del clasificador por planta	PEC3	Completed
Desarrollo y finalización de función de clasificación de habitación según clu	PEC3	Completed
Redacción	PEC3	In progress
Reporte de Clustering	PEC3	Completed
Finalización y reporte de resultados según el mejor valor de k para la predic	PEC3	Completed
Entrega PEC 3	PEC3	In progress
Entrega PEC 4.1	PEC4.1	Not started
Entrega PEC 4.2	PEC4.2	Not started
Preparación defensa del proyecto	Defensa	Not started
Defensa pública y presentación	Defensa	Not started
Diseño de enfoque básico de Incremental Learning - Spark		Not started
Sistema básico de envío de información por timestamp en batches - Spark		Not started
Diseño enfoque básico de Transfer Learning		Not started
Utilización de Carto - Crear mapa en predicciones		Not started
Bibliografía en app del trabajo		Not started
+ New		
COUNT 70		

Figura B.7: Workflow - Sección 7

Timeline tasks

Name	Date	Status	Tags
Resumen de la propuesta - abstract	@February 16, 2022 → February 18, 2022	Completed	PEC1
Palabras clave	@February 16, 2022 → February 16, 2022	Completed	PEC1
Propuesta de título	@February 16, 2022 → February 16, 2022	Completed	PEC1
Descripción, justificación del interés y relevancia de la propuesta	@February 18, 2022 → February 22, 2022	Completed	PEC1
Explicación de la motivación personal	@February 22, 2022 → February 23, 2022	Completed	PEC1
Definición de objetivos	@February 23, 2022 → February 24, 2022	Completed	PEC1
Descripción de metodología empleada	@February 24, 2022 → February 25, 2022	Completed	PEC1
Planificación o plan de investigación	@February 25, 2022 → February 26, 2022	Completed	PEC1
Entrega PEC 1	@February 27, 2022	Completed	PEC1
Bibliografía	@February 27, 2022 → February 27, 2022	Completed	PEC1
Contexto e historia	@February 28, 2022 → March 1, 2022	Completed	PEC2
Tecnologías IPS - Técnicas cálculo de posición	@March 1, 2022 → March 2, 2022	Completed	PEC2
Tecnologías IPS - Wi-Fi	@March 2, 2022 → March 3, 2022	Completed	PEC2
Tecnologías IPS - BLE	@March 3, 2022 → March 4, 2022	Completed	PEC2
Tecnologías IPS - UWB	@March 4, 2022 → March 4, 2022	Completed	PEC2
Aplicaciones - Proximidad	@March 4, 2022 → March 5, 2022	Completed	PEC2
Aplicaciones - Posicionamiento	@March 5, 2022 → March 6, 2022	Completed	PEC2
Localización mediante Wi-Fi RSS Fingerprinting	@March 6, 2022 → March 8, 2022	Completed	PEC2
Localización mediante BLE	@March 8, 2022 → March 10, 2022	Completed	PEC2
Modelado - Algoritmo k-NN y SVM	@March 10, 2022 → March 10, 2022	Completed	PEC2
Modelado - Clustering	@March 10, 2022 → March 11, 2022	Completed	PEC2
Modelado - Incremental Learning	@March 11, 2022 → March 12, 2022	Completed	PEC2
Modelado - Transfer Learning	@March 11, 2022 → March 11, 2022	Completed	PEC2
Modelado - Ensemble models	@March 12, 2022 → March 12, 2022	Completed	PEC2
Bibliografía	@March 12, 2022 → March 13, 2022	Completed	PEC2
Entrega PEC2	@March 13, 2022	Completed	PEC2
Research sobre infraestructura	@March 14, 2022 → March 17, 2022	Completed	PEC3
Puesta en marcha del repositorio GIT	@March 17, 2022 → March 18, 2022	Completed	PEC3
Puesta en marcha del entorno con anaconda	@March 18, 2022 → March 18, 2022	Completed	PEC3
Primer prototipo de Jupyter notebooks	@March 18, 2022 → March 19, 2022	Completed	PEC3
Puesta en marcha del framework Ploomber	@March 19, 2022 → March 22, 2022	Completed	PEC3
Puesta en marcha de MLFlow	@March 22, 2022 → March 25, 2022	Completed	PEC3
Primer prototipo de la etapa de reporte	@March 25, 2022 → March 27, 2022	Completed	PEC3
Primer prototipo de la etapa de limpieza de datos	@March 27, 2022 → March 30, 2022	Completed	PEC3
Primer prototipo de etapa de entrenamiento y clustering	@March 30, 2022 → April 3, 2022	Completed	PEC3

Name	Date	Status	Tags
Primer prototipo de visualización	@April 3, 2022 → April 7, 2022	Completed	PEC3
Mejora del diagrama de etapas	@April 7, 2022 → April 9, 2022	Completed	PEC3
Implementación sistema Logging	@April 9, 2022 → April 11, 2022	Completed	PEC3
Entrenar modelo Clustering y realizar predicciones	@April 11, 2022 → April 12, 2022	Completed	PEC3
Separar dataframe por clusters	@April 11, 2022 → April 12, 2022	Completed	PEC3
Group by z - planta o altura previo a Clustering	@April 11, 2022 → April 12, 2022	Completed	PEC3
Comprobar el por qué en ciertos clusters aparecen datos raros de accuracy	@April 12, 2022 → April 14, 2022	Completed	PEC3
Nested Cross Validation	@April 12, 2022 → April 21, 2022	Completed	PEC3
A cada uno de los clusters aplicar kNN	@April 12, 2022 → April 15, 2022	Completed	PEC3
Crear visualización actualizada de Pipeline físico → digital	@April 12, 2022 → April 13, 2022	Completed	PEC3
Limpieza de datos más efectiva	@April 13, 2022 → April 21, 2022	Completed	PEC3
Cálculo de errores	@April 14, 2022 → April 21, 2022	Completed	PEC3
Paralelizar algoritmos en etapas - .yml	@April 16, 2022 → April 22, 2022	Completed	PEC3
Investigar más sobre MLFlow para sacarle el máximo jugo	@April 16, 2022 → April 22, 2022	Completed	PEC3
PCA en Clustering	@April 18, 2022 → April 22, 2022	Completed	PEC3
Paralelizar datasets - .yml	@April 20, 2022 → April 24, 2022	Completed	PEC3
Finalización etapa de Clustering con KMeans	@April 21, 2022 → April 29, 2022	Completed	PEC3
Mejora y finalización del preprocesado de datos en ficheros .npy	@April 21, 2022 → April 22, 2022	Completed	PEC3
Generalizar pipeline para utilizar muchos datasets	@April 21, 2022 → April 25, 2022	Completed	PEC3
Finalización etapa del clasificador por edificio	@April 22, 2022 → April 29, 2022	Completed	PEC3
Finalización etapa del clasificador por planta	@April 23, 2022 → April 30, 2022	Completed	PEC3
Desarrollo y finalización de función de clasificación de habitación según clusters	@April 30, 2022 → May 11, 2022	Completed	PEC3
Redacción	@May 1, 2022 → May 15, 2022	In progress	PEC3
Reporte de Clustering	@May 9, 2022 → May 12, 2022	Completed	PEC3
Finalización y reporte de resultados según el mejor valor de k para la predicción por Clúster	@May 10, 2022 → May 12, 2022	Completed	PEC3
Entrega PEC 3	@May 15, 2022	In progress	PEC3
Entrega PEC 4.1	@May 29, 2022 → May 29, 2022	Not started	PEC4.1
Entrega PEC 4.2	@June 5, 2022	Not started	PEC4.2
Preparación defensa del proyecto	@June 6, 2022 → June 12, 2022	Not started	Defensa
Defensa pública y presentación	@June 13, 2022 → June 24, 2022	Not started	Defensa

Apéndice C

Anaconda, Python e instalación de librerías asociadas

Primero será necesario clonar el repositorio localmente con:

```
git clone git@github.com:yinx89/ips-algorithms-study.git
```

A continuación, se pasa a la creación del entorno de anaconda usando Python 3.7:

```
conda create -n general python=3.7
```

Podemos ver qué entornos hemos creado anteriormente y activar cada uno de ellos con estos comandos:

```
conda info --envs
conda activate name/path
```

A continuación, instalamos varias de las librerías necesarias:

```
pip install ploomber
pip install mlflow
pip install pandas
/usr/local/bin/brew install libomp
pip install scikit-learn==0.23.2
conda install scikit-learn==0.23.2
conda install NumPy=1.20
pip install pygraphviz<1.8
conda install pydot-ng
conda install graphviz
```

Finalmente, para poder reproducir posteriormente un entorno idéntico se exporta a un fichero .yml de la siguiente manera:

```
conda env export > environment.yml
```

Apéndice D

Instalación y uso de Ploomber & MLFlow

Para empezar a usar ambas herramientas, en primer lugar se crea la estructura de directorio de Ploomber:

```
ploomber scaffold sample-ml --empty
```

A continuación, se instalan las dependencias de environment.yml:

```
ploomber install
```

Para probar MLFlow localmente se puede realizar la siguiente prueba:

```
cd sample-ml/mlflow
mlflow server --backend-store-uri sqlite:///ips.db
        --default-artifact-root file:./mlruns

[2022-03-24 22:49:23 +0100] [2941] [INFO]
        Starting gunicorn 20.1.0
[2022-03-24 22:49:23 +0100] [2941] [INFO]
        Listening at: http://127.0.0.1:5000 (2941)
```

De esta manera iniciamos el servidor en una instancia local utilizando como backend-store una base de datos sqlite y utilizando una ruta por defecto donde almacenar los artefactos a través de la carpeta mlruns. De esta manera todo lo que registráramos en nuestro código en cada componente escrito en Python, quedaría registrado en esta instancia, para posteriormente poder visualizarlo en una plataforma HTML en localhost. Vemos que en segundo plano, gunicorn permite mostrar la información en una plataforma web. Posteriormente se usará en el proyecto el mismo enfoque de MLFlow pero ejecutándolo en una instancia de Google Cloud. Para la creación de los notebooks en Ploomber se usa:

```
ploomber scaffold
```

Para que cada componente registre los parámetros de entrada del fichero .yml (es decir, el comando a ejecutar cada vez que registremos nuevos valores en el fichero .yml) es el siguiente:

```
ploomber nb --inject
```

Para mostrar el pipeline asociado a ese fichero .yml según cada componente se ejecuta:

```
ploomber plot
```

Y finalmente, para ejecutar el pipeline por componentes se ejecuta el siguiente comando:

```
ploomber build
```

Se debe tener en cuenta que esto lo que hace es ejecutar cada una de las tareas que han sido definidas en el fichero .yml que corresponde a cada una de las tareas, de manera secuencial o en paralelo (dependiendo de cómo se haya definido el fichero .yml) [13]. Se debe tener en cuenta también que si el framework detecta que en el pipeline no hay cambios, solo empezará la ejecución en etapas o puntos desde donde detecte que haya cambios. Así evitamos tener que, por ejemplo, limpiar los datos de nuevo aunque no se hayan realizado cambios en este componente. Finalmente, se puede forzar a ejecutar desde el inicio el pipeline con:

```
ploomber build --force
```

Apéndice E

Informe y registro de eventos

```
2022-04-17 01:53:45,223
        - __main__ - DEBUG - ----------------------------------------
2022-04-17 01:53:45,223
        - __main__ - DEBUG - MAIN - Start of the ETL process --
2022-04-17 01:53:45,225
        - __main__ - DEBUG - ----------------------------------------
2022-04-17 01:53:45,225
        - __main__ - DEBUG - buffer created                        --
2022-04-17 01:53:45,226
        - __main__ - DEBUG - ----------------------------------------
2022-04-17 01:53:46,514
        - __main__ - DEBUG - Files read:                           --
2022-04-17 01:53:46,515
        - __main__ - DEBUG -
        /Users/gines/ips-algorithms-study/input_data/UJI1_trncrd.csv
2022-04-17 01:53:46,516
        - __main__ - DEBUG -
        /Users/gines/ips-algorithms-study/input_data/UJI1_trnrss.csv
2022-04-17 01:53:46,517
        - __main__ - DEBUG -
        /Users/gines/ips-algorithms-study/input_data/UJI1_tstcrd.csv
2022-04-17 01:53:46,517
        - __main__ - DEBUG -
        /Users/gines/ips-algorithms-study/input_data/UJI1_tstrss.csv
2022-04-17 01:53:46,518
```

```
        - __main__ - DEBUG - ----------------------------------------
2022-04-17 01:53:46,518
        - __main__ - DEBUG - Dataframe first/last rows:        ---
2022-04-17 01:53:46,519
        - __main__ - DEBUG - ----------------------------------------
        longitud       latitud  altura  planta  edificio
0      -7541.2643  4.864921e+06     7.4       2         1
...           ...           ...     ...     ...       ...
19860  -7536.1658  4.864898e+06    11.1       3         1

[19861 rows x 5 columns]
       WAP001  WAP002  WAP003  WAP004  WAP005  WAP006  WAP007  WAP008  WAP
0         100     100     100     100     100     100     100     100    1
...       ...     ...     ...     ...     ...     ...     ...     ...
19860     100     100     100     100     100     100     100     100    1
2022-04-17 01:53:46,556
        - __main__ - DEBUG - ----------------------------------------
2022-04-17 01:53:46,558
        - __main__ - DEBUG - Dataframe info from buffer:       ---
2022-04-17 01:53:46,558
        - __main__ - DEBUG - ----------------------------------------
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19861 entries, 0 to 19860
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   longitud   19861 non-null  float64
 1   latitud    19861 non-null  float64
 2   altura     19861 non-null  float64
 3   planta     19861 non-null  int64
 4   edificio   19861 non-null  int64
 ----------------------------------------
Values from columns:              ---
----------------------------------------
WAP488
 100     1110
```

```
−93          1
Name: WAP488, dtype: int64
```

WAP235

```
100      1111
Name: WAP235, dtype: int64
```

Apéndice F

Diagrama y flujo de datos

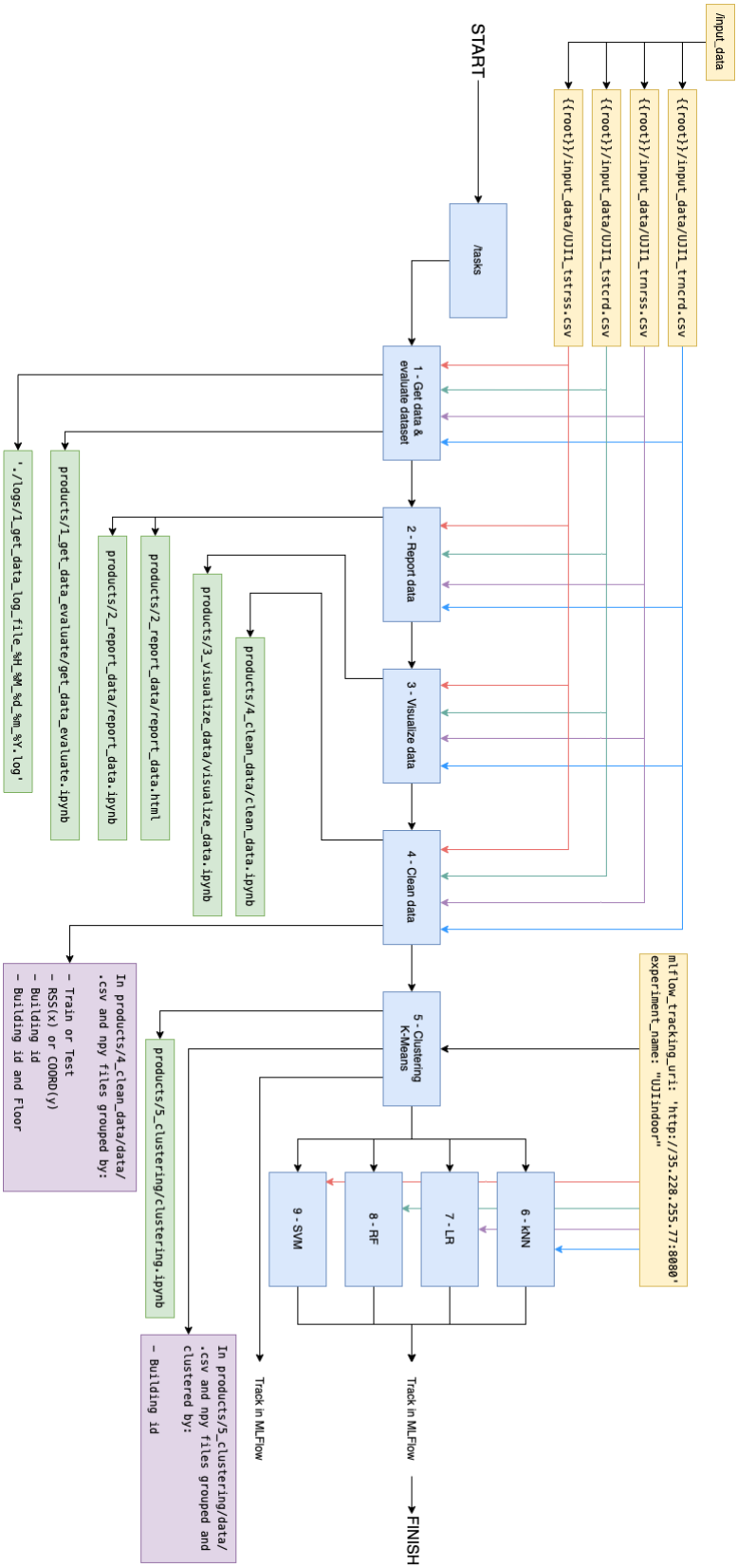
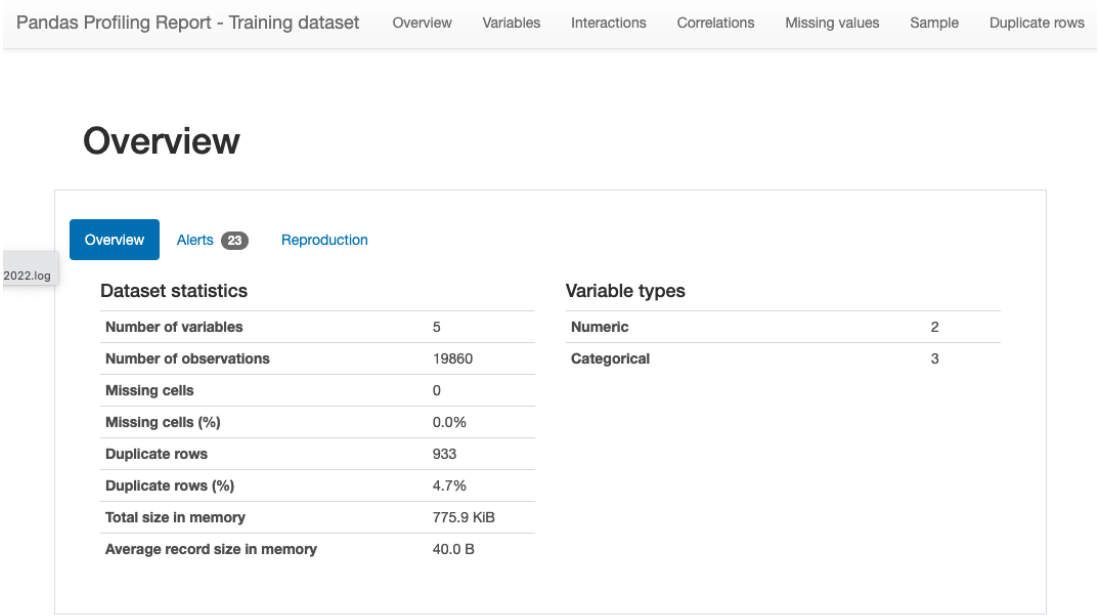


Figura F.1: Diagrama global de funcionamiento del sistema

Apéndice G

Informe sobre el dataset

En primer lugar se muestra el panorama general:



Pandas Profiling Report - Training dataset | Overview | Variables | Interactions | Correlations | Missing values | Sample | Duplicate rows

Overview

Overview | Alerts 23 | Reproduction

2022.log

Dataset statistics

Number of variables	5
Number of observations	19860
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	933
Duplicate rows (%)	4.7%
Total size in memory	775.9 KiB
Average record size in memory	40.0 B

Variable types

Numeric	2
Categorical	3

Figura G.1: Overview Pandas Profiling Report

En segundo lugar, se observan algunas alertas que muestran posibles altas correlaciones entre variables y valores duplicados. Este archivo en concreto no es muy aclaratorio por la misma naturaleza de las coordenadas, pero con el fichero de potencias en los puntos de acceso se puede encontrar posibles valores altamente correlacionados.

Pandas Profiling Report - Training dataset | Overview | Variables | Interactions | Correlations | Missing values | Sample | Duplicate rows

Alerts

Dataset has 933 (4.7%) duplicate rows	Duplicates
-7541.264299999923 is highly correlated with 4864920.778200001 and 1 other fields	High correlation
4864920.778200001 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
7.4 is highly correlated with 2	High correlation
2 is highly correlated with 7.4	High correlation
1 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
-7541.264299999923 is highly correlated with 4864920.778200001 and 1 other fields	High correlation
4864920.778200001 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
7.4 is highly correlated with 2	High correlation
2 is highly correlated with 7.4	High correlation
1 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
-7541.264299999923 is highly correlated with 4864920.778200001 and 1 other fields	High correlation
4864920.778200001 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
7.4 is highly correlated with 2	High correlation
2 is highly correlated with 7.4	High correlation
1 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
2 is highly correlated with 7.4	High correlation
7.4 is highly correlated with 2	High correlation
-7541.264299999923 is highly correlated with 4864920.778200001 and 1 other fields	High correlation
4864920.778200001 is highly correlated with -7541.264299999923 and 1 other fields	High correlation
7.4 is highly correlated with 2	High correlation
2 is highly correlated with 7.4	High correlation
1 is highly correlated with -7541.264299999923 and 1 other fields	High correlation

Figura G.2: Alerts Pandas Profiling Report

En tercer lugar, aparece información sobre cuando se ha realizado el análisis, la duración, version de la librería y la posibilidad de descargar la configuración. Cabe destacar que este reporte puede ser personalizado y añadido nuevas funcionalidades que resulten interesantes para este tipo de información.

Overview

Overview | Alerts 23 | Reproduction

Reproduction

Analysis started	2022-04-25 15:25:19.311591
Analysis finished	2022-04-25 15:25:23.619072
Duration	4.31 seconds
Software version	pandas-profiling v3.1.0
Download configuration	config.json

Figura G.3: Reproduction Pandas Profiling Report

En cuarto lugar, se puede observar un análisis variable a variable. Se puede pulsar 'Toggle details' para obtener más información estadística, el histograma en versión completa, valores comunes y valores extremos:

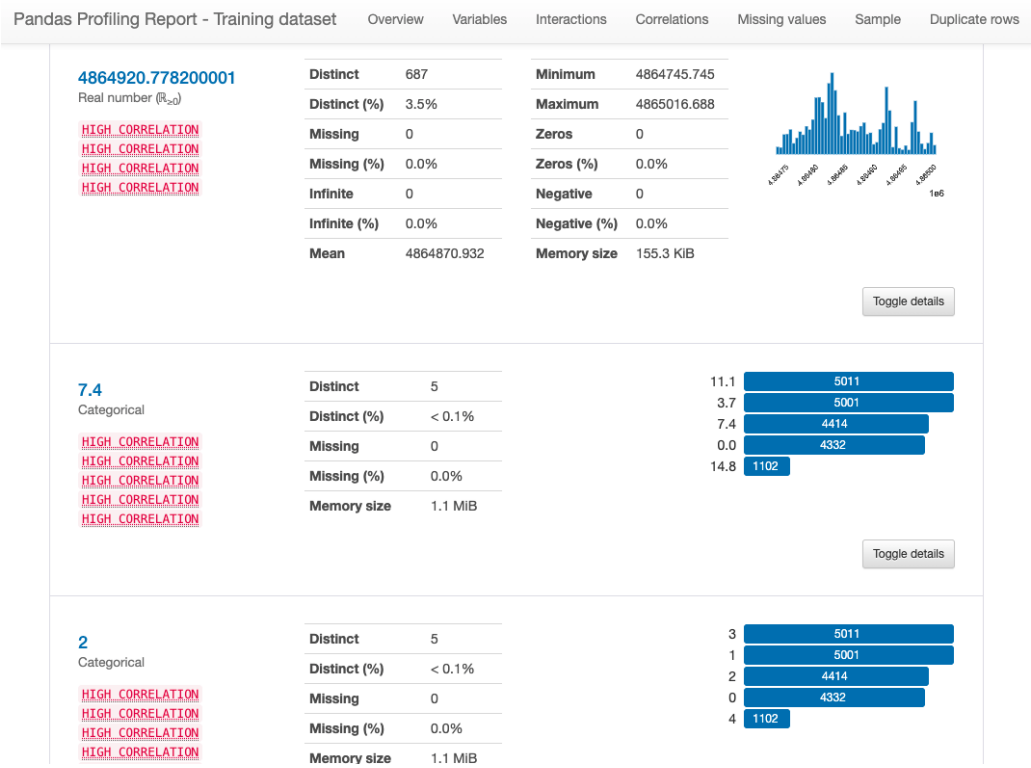


Figura G.4: Variables Pandas Profiling Report

En la sección de Interacciones se puede observar un mapa que muestra las variables de latitud y longitud y la densidad de puntos en secciones hexagonales. Para valores geográficos se puede ver cómo estarían distribuidas las variables, como es el caso.

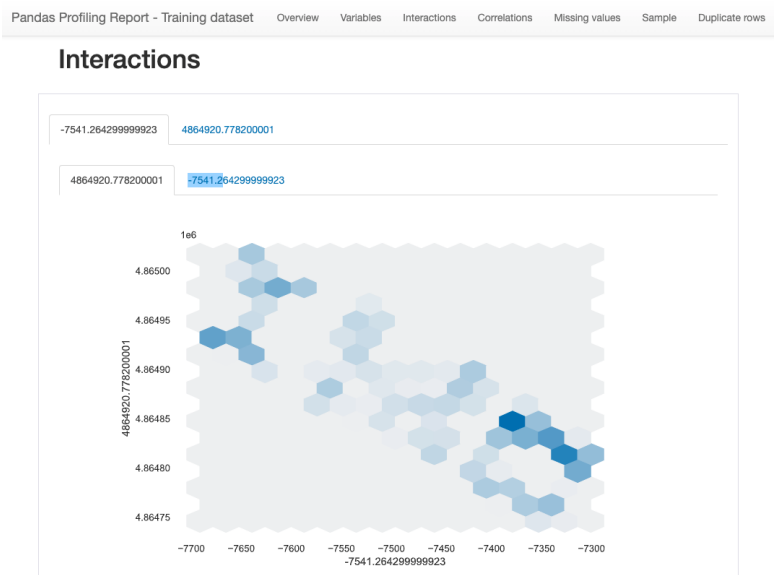


Figura G.5: Interactions Pandas Profiling Report

En la sección Correlaciones se muestra la correlación entre variables atendiendo a distintos métodos:

Figura G.6: Correlations Pandas Profiling Report

En la siguiente sección se muestran los valores que faltan por variables:

Figura G.7: Missing Values Pandas Profiling Report

En la penúltima sección se muestran los primeros y últimos registros del dataset:

Pandas Profiling Report - Training dataset COORD

Overview Variables Interactions Correlations Missing values Sample Duplicate rows

Sample

First rows

	-7541.264299999923	4864920.778200001	7.4	2	1
0	-7536.621200	4.864934e+06	7.4	2	1
1	-7519.152400	4.864950e+06	7.4	2	1
2	-7524.570400	4.864934e+06	7.4	2	1
3	-7533.896200	4.864939e+06	7.4	2	1
4	-7519.152400	4.864950e+06	7.4	2	1
5	-7527.451100	4.864929e+06	7.4	2	1
6	-7559.497300	4.864888e+06	7.4	2	1
7	-7510.437173	4.864949e+06	7.4	2	1
8	-7528.816402	4.864959e+06	7.4	2	1
9	-7523.628200	4.864952e+06	7.4	2	1

Figura G.8: Sample Pandas Profiling Report

Por último, se muestran los valores duplicados ordenados por mayor número de frecuencia en el dataset:

Pandas Profiling Report - Training dataset COORD

Overview Variables Interactions Correlations Missing values Sample Duplicate rows

Duplicate rows

Most frequently occurring

	-7541.264299999923	4864920.778200001	7.4	2	1	# duplicates
452	-7445.557874	4.864827e+06	0.0	0	1	80
418	-7468.870245	4.864828e+06	0.0	0	1	70
286	-7549.379435	4.864878e+06	3.7	1	1	62
292	-7541.168800	4.864921e+06	3.7	1	1	60
719	-7360.539300	4.864838e+06	0.0	0	2	60
888	-7316.835800	4.864814e+06	0.0	0	2	60
731	-7359.297900	4.864837e+06	0.0	0	2	58
818	-7333.069100	4.864769e+06	0.0	0	2	54
273	-7560.377608	4.864859e+06	3.7	1	1	50
309	-7535.209515	4.864948e+06	3.7	1	1	50

Figura G.9: Duplicate Rows Pandas Profiling Report

Apéndice H

Plataforma MLFlow

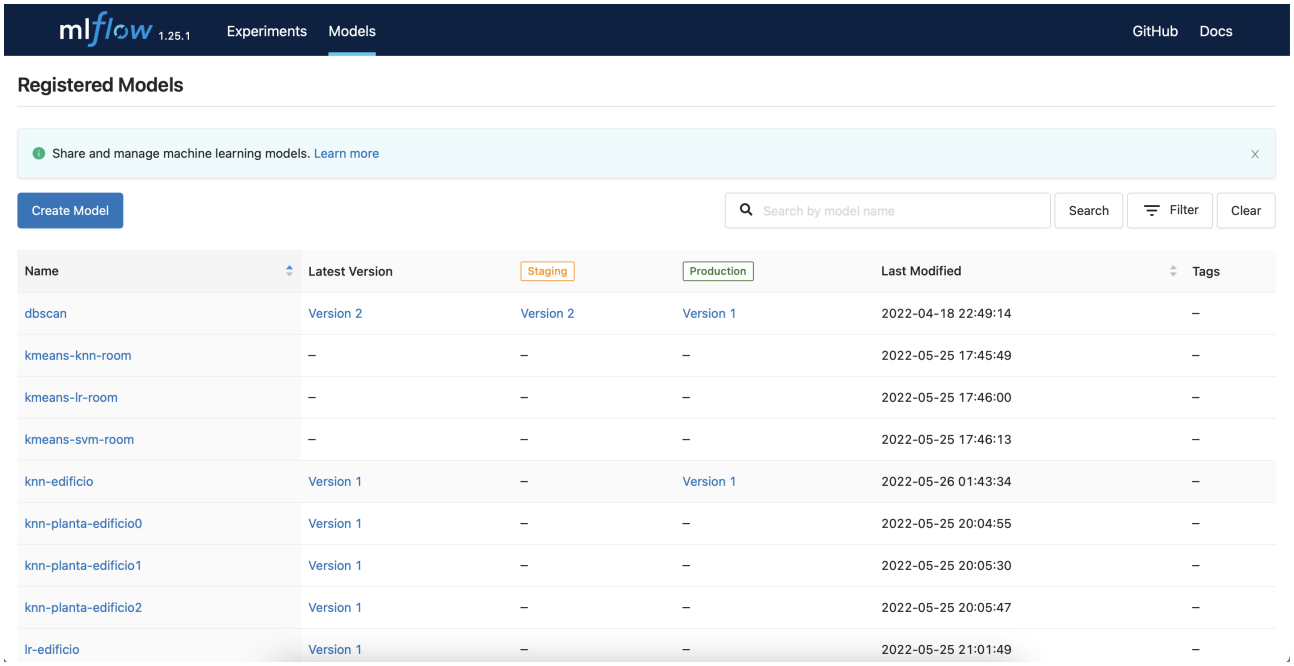


Figura H.1: Pantalla principal de modelos - MLFlow

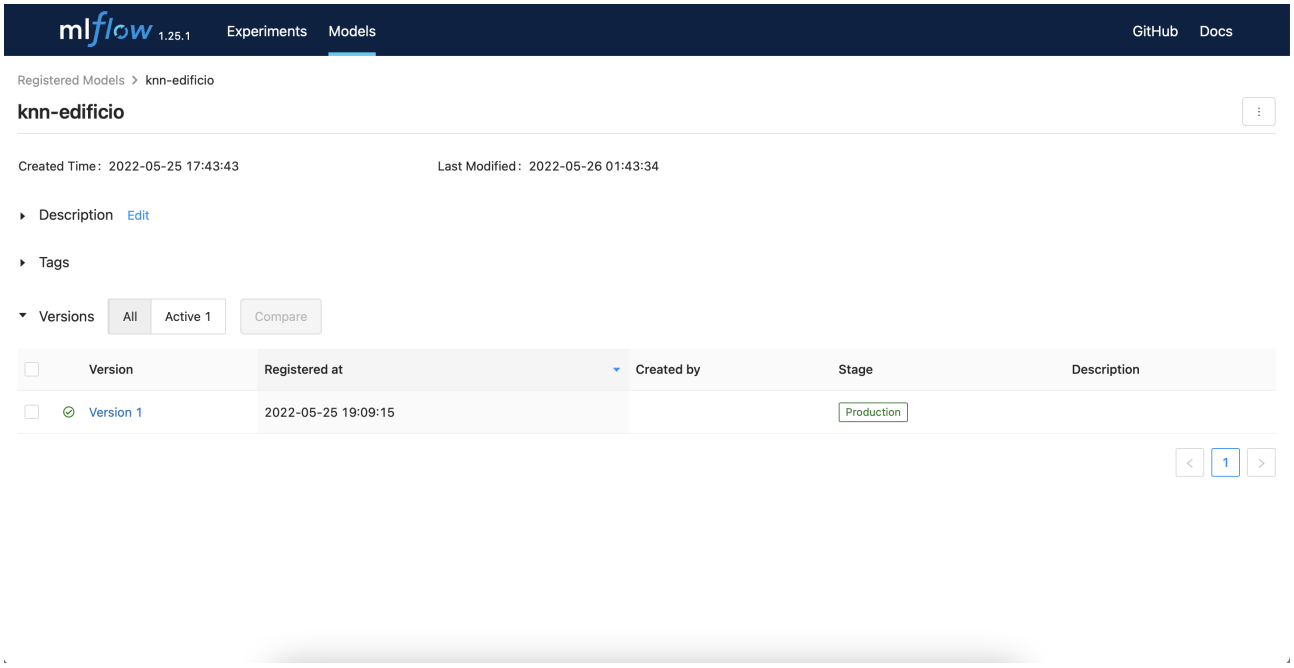


Figura H.2: Vista del modelo en específico - MLFlow

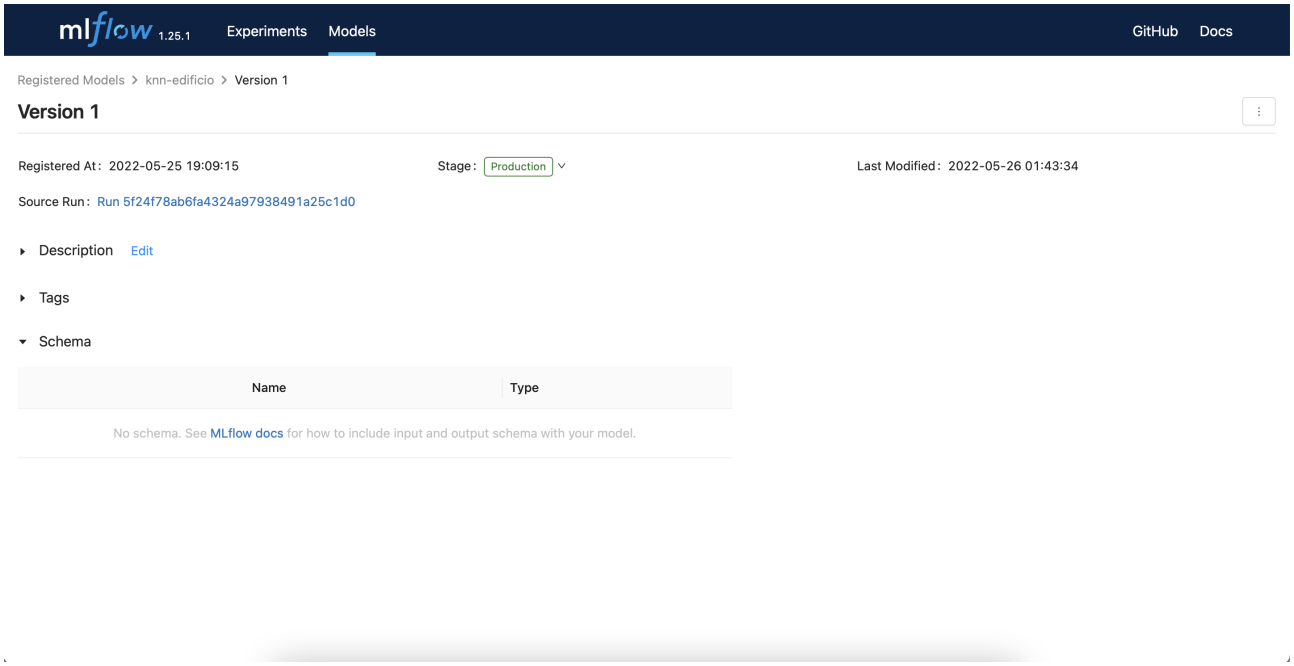


Figura H.3: Vista de la Versión del modelo - MLFlow

mlflow 1.25.1 Experiments Models GitHub Docs

knn-edificio > Run 5f24f78ab6fa4324a97938491a25c1d0

Run 5f24f78ab6fa4324a97938491a25c1d0

Date: 2022-05-25 19:00:41 | Source: ipykernel_launcher.py | User: gines

Duration: 6.6min | Status: FINISHED | Lifecycle Stage: active

- Description Edit
- Parameters
- Metrics (5)
- Tags
- Artifacts

knn-edificio
- MLmodel
- conda.yaml
- model.pkl
- requirements.txt

Full Path:gs://example_bucket_data_science_1/7/5f24f78ab6fa4324a97938491a25c1d0/artifacts/knn-edificio

knn-edificio, v1
Registered on 2022/05/25

MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. This model is also registered to the model registry.

Model schema

Input and output schema for your model. Learn more

Make Predictions

Predict on a Spark DataFrame:

Figura H.4: Detalles de la Versión del modelo - MLFlow

knn-edificio
- MLmodel
- conda.yaml
- model.pkl
- requirements.txt

Full Path:gs://example_bucket_data_science_1/7/5f24f78ab6fa4324a97938491a25c1d0/artifacts/knn-edificio

knn-edificio, v1
Registered on 2022/05/25

MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. This model is also registered to the model registry.

Model schema

Input and output schema for your model. Learn more

Name	Type

No schema. See MLflow docs for how to include input and output schema with your model.

Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
logged_model = 'runs:/5f24f78ab6fa4324a97938491a25c1d0/knn-edificio'

# Load model as a Spark UDF. Override result_type if the model does not
return double values.
loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=logged_model, re
sult_type='double')

# Predict on a Spark DataFrame.
columns = list(df.columns)
df.withColumn('predictions', loaded_model(*columns)).collect()
```

Predict on a Pandas DataFrame:

```
import mlflow
logged_model = 'runs:/5f24f78ab6fa4324a97938491a25c1d0/knn-edificio'

# Load model as a PyFuncModel.
loaded_model = mlflow.pyfunc.load_model(logged_model)

# Predict on a Pandas DataFrame.
import pandas as pd
loaded_model.predict(pd.DataFrame(data))
```

Figura H.5: Detalles de la versión del modelo - Predicciones

Bibliografía

[1] Amos Kingatua. Bluetooth Indoor Positioning and Tracking Solutions. https://medium.com/supplyframe-hardware/bluetooth-indoor-positioning-and-asset-tracking-solutions-8c78cae0a03. Accessed: 13-03-2022.

[2] Ander Fernández Jauregui. Tutorial MLflow: MLOps de forma sencilla. https://anderfernandez.com/blog/tutorial-mlflow-completo/. Accessed: 15-05-2022.

[3] Ahmed Azeez, Saba Jabbar, Dr.Mohammed Sulttan, Desheng Wang, and Medo Toty. Wireless indoor localization systems and techniques: Survey and comparative study. *Indonesian Journal of Electrical Engineering and Computer Science*, 3:392–409, 08 2016.

[4] Ahmed Azeez, Saba Jabbar, Dr.Mohammed Sulttan, Desheng Wang, and Medo Toty. Wireless indoor localization systems and techniques: Survey and comparative study. *Indonesian Journal of Electrical Engineering and Computer Science*, 3:392–409, 08 2016.

[5] Vladimir Bellavista-Parent, Joaquín Torres-Sospedra, and Antoni Perez-Navarro. New trends in indoor positioning based on wifi and machine learning: A systematic review. In *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, 2021.

[6] Dibya Bora and Dr Gupta. Effect of different distance measures on the performance of k-means algorithm: An experimental study in matlab. *arXiv*, 5, 05 2014.

[7] Alexis Richard Claridades and Jiyeong Lee. Developing a data model of indoor points of interest to support location-based services. *Journal of Sensors*, 2020:1–16, 08 2020.

[8] Confederación Plena inclusión España. Wayfinding. https://www.plenainclusion.org/discapacidad-intelectual/recurso/wayfinding/. Accessed: 13-03-2022.

[9] Kevin Curran, Eoghan Furey, Tom Lunney, Jose Santos, Derek Woods, and Aidan Mccaughey. An evaluation of indoor location determination technologies. *J. Location Based Services*, 5:61–78, 06 2011.

[10] Álvaro De-La-Llana-Calvo, José Lázaro, Alfredo Gardel, David Rodriguez, Ignacio Bravo, and Felipe Zapata. Sistema de posicionamiento en interiores basado en iluminación led y sensor psd. In *Conference: Seminario Anual de Automática, Electrónica industrial e Instrumentación (SAAEI19)At: Córdoba, España*, 07 2019.

[11] Denise Trevisan Troche. Influencia de la presencia de personas en sistemas de posicionamiento indoor mediante Wi-Fi fingerprinting. http://hdl.handle.net/10609/58945. Accessed: 13-03-2022.

[12] Eduardo Blancas. Ploomber: Maintainable and Collaborative Pipelines in Jupyter. https://blog.jupyter.org/ploomber-maintainable-and-collaborative-pipelines-in-jupyter-acb3ad2101a7. Accessed: 15-05-2022.

[13] Eduardo Blancas. Refactoring a Jupyter notebook into a maintainable pipeline: A step-by-step guide (Part I). https://ploomber.io/blog/refactor-nb-i/. Accessed: 15-05-2022.

[14] Fernando Sancho Caparrini. Algoritmos de Clustering. http://www.cs.us.es/~fsancho/?e=230. Accessed: 13-03-2022.

[15] Pedro J. García-Paterna, Alejandro S. Martínez-Sala, and Juan Carlos Sánchez-Aarnoutse. Empirical study of a room-level localization system based on bluetooth low energy beacons. *Sensors*, 21(11), 2021.

[16] Alexander Gepperth and Barbara Hammer. Incremental learning algorithms and applications. In *European Symposium on Artificial Neural Networks (ESANN)*, Bruges, Belgium, 2016.

[17] Gilbert Tanner. Density-Based Spatial Clustering of Applications with Noise (DBSCAN). https://ml-explained.com/blog/dbscan-explained. Accessed: 05-06-2022.

[18] Henrik Skogström. What Is The Difference Between DevOps And MLOps? https://valohai.com/blog/difference-between-devops-and-mlops/. Accessed: 15-05-2022.

[19] Ismael Miranda Gordo. Fusión de información de sistemas de localización para la evaluación de rutinas de comportamiento de personas. https://ebuah.uah.es/xmlui/bitstream/handle/10017/49636/TFM_Miranda_Gordo_2021.pdf?sequence=1&isAllowed=y. Accessed: 13-03-2022.

[20] ISO Central Secretary. Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary — Part 5: Locating systems. Standard, International Standards Organization (ISO), Geneva, CH, June 2008.

[21] Kang Eun Jeon, James She, Perm Soonsawad, and Pai Chet Ng. Ble beacons for internet of things applications: Survey, challenges, and opportunities. *IEEE Internet of Things Journal*, 5(2):811–828, 2018.

[22] Joaquín Amat Rodrigo. Máquinas de Vector Soporte (SVM) con Python. https://www.cienciadedatos.net/documentos/py24-svm-python.html. Accessed: 29-05-2022.

[23] José M. Llamas. Sistemas de geolocalización en interiores. https://blog.cartif.es/sistemas-de-geolocalizacion-en-interiores/. Accessed: 13-03-2022.

[24] Junta de Castilla y León - ORSI. Sistemas de localización e información geográfica. https://issuu.com/makketa./docs/sistemas_de_localizacion_jcyl. Accessed: 13-03-2022.

[25] Roman Klus, Lucie Klus, Jukka Talvitie, Jaakko Pihlajasalo, Joaquín Torres-Sospedra, and Mikko Valkama. Transfer learning for convolutional indoor positioning systems. In *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, 2021.

[26] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. 14, 03 2001.

[27] Toshiaki Koike-Akino, Pu Wang, Milutin Pajovic, Haijian Sun, and Philip V. Orlik. Fingerprinting-based indoor localization with commercial mmwave wifi: A deep learning approach. *IEEE Access*, 8:84879–84892, 2020.

[28] Hangxin Liu, Zeyu Zhang, Yixin Zhu, and Song-Chun Zhu. Self-supervised incremental learning for sound source localization in complex indoor environment. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2599–2605, 2019.

[29] Luis Díaz-Ambrona Tabernilla. Trabajo fin de carrera - Sistema de localización en interiores. https://oa.upm.es/947/1/PFC_LUIS_DIAZ_AMBRONA.pdf. Accessed: 13-03-2022.

[30] Marc Compte - UNIGIS. 09 Jul 2018 Posicionamiento Indoor. https://www.unigis.es/posicionamiento-indoor/. Accessed: 13-03-2022.

[31] Microsoft. RADAR. https://www.microsoft.com/en-us/research/project/radar/. Accessed: 13-03-2022.

[32] MLFlow. MLflow Model Registry. https://www.mlflow.org/docs/latest/model-registry.html. Accessed: 04-06-2022.

[33] Médéric MOREL. How does Indoor Positioning work? https://www.mapwize.io/blog/how-does-indoor-positioning-work/. Accessed: 13-03-2022.

[34] Dmitry Namiot. On indoor positioning. *International Journal of Open Information Technologies*, 3:23–26, 03 2015.

[35] J Pancham, Richard Millham, and Simon James Fong. Evaluation of real time location system technologies in the health care sector. In *2017 17th International Conference on Computational Science and Its Applications (ICCSA)*, pages 1–7, 2017.

[36] Doan Tinh Pham and Ta Thi Ngoc Mai. Ensemble learning model for wifi indoor positioning systems. *IAES International Journal of Artificial Intelligence*, 10:200–206, 2021.

[37] Alwin Poulose, Jihun Kim, and Dong Seog Han. A sensor fusion framework for indoor localization using smartphone sensors and wi-fi rssi measurements. *Applied Sciences*, 9(20), 2019.

[38] Michael Quan, Eduardo Navarro, and Benjamin Peuker. Wi-fi localization using rssi fingerprinting. *Computer Engineering*, 01 2010.

[39] Darwin Quezada-Gaibor, Lucie Klus, Joaquiín Torres-Sospedra, Elena Simona Lohan, Jari Nurmi, and Joaquín Huerta. Improving dbscan for indoor positioning using wi-fi radio maps in wearable and iot devices. In *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 208–213, 2020.

[40] Darwin Quezada-Gaibor, Lucie Klus, Joaquín Torres-Sospedra, Elena Simona Lohan, Jari Nurmi, and Joaquín Huerta. Improving dbscan for indoor positioning using wi-fi radio maps in wearable and iot devices. In *Conference: 2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 10 2020.

[41] Rashida Nasrin Sucky. Multiclass Classification Using Logistic Regression from Scratch in Python: Step by Step Guide. https://towardsdatascience.com/multiclass-classification-algorithm-from-scratch-with-a-project-in-python-step-by
Accessed: 29-05-2022.

[42] Cristina Rodriguez-Martinez and Joaquín Torres-Sospedra. Revisiting the analysis of hyperparameters in k-nn for wi-fi and ble fingerprinting: Current status and general results.

In *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, 2021.

[43] Samuel Fernández. A finales de 2018 habrá 3.000 millones de usuarios de smartphones en el mundo, según Newzoo. https://www.xatakamovil.com/mercado/a-finales-2018-habra-3-000-millones-usuarios-smartphones-mundo-newzoo. Accessed: 13-03-2022.

[44] Samuel Fernández. A finales de 2018 habrá 3.000 millones de usuarios de smartphones en el mundo, según Newzoo. https://www.comunicacionesinalambricashoy.com/wireless/las-nuevas-frecuencias-de-wifi/. Accessed: 13-03-2022.

[45] Sciforce. MLOps: Comprehensive Beginner's Guide. https://medium.com/sciforce/mlops-comprehensive-beginners-guide-c235c77f407f. Accessed: 14-05-2022.

[46] D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. Hidden technical debt in machine learning systems. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2503–2511, Cambridge, MA, USA, 2015. MIT Press.

[47] Ahmet Çağdaş Seçkin and Aysun Coşkun. Hierarchical fusion of machine learning algorithms in indoor positioning and localization. *Applied Sciences*, 9(18), 2019.

[48] Smithsonian - National Air and Space Museum. A New Era in Time and Navigation. https://timeandnavigation.si.edu/navigating-air/navigation-at-war/new-era-in-time-and-navigation. Accessed: 13-03-2022.

[49] Sonia Lara Cordón. Posicionamiento en interiores con sensores inalámbricos. https://oa.upm.es/67025/1/TFM_SONIA_LARA_CORDON.pdf. Accessed: 13-03-2022.

[50] Srce Cde. MLOps Explained. https://medium.com/practical-data-science-and-engineering/mlops-explained-a8b2475e5e42. Accessed: 14-05-2022.

[51] TI DevTools. Bluetooth Low Energy Scanning and Advertising. https://dev.ti.com/tirex/explore/node?node=AD4sGbaamTCyn0DvZgBAsg__krol.2c__LATEST. Accessed: 13-03-2022.

[52] Tristan Roberts. We Spend 90 % of Our Time Indoors. Says Who? https://www.buildinggreen.com/blog/we-spend-90-our-time-indoors-says-who. Accessed: 13-03-2022.

[53] Vibhu Varshney, Rajat Kant Goel, and Mohammed Abdul Qadeer. Indoor positioning system using wi-fi amp; bluetooth low energy technology. In *2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN)*, pages 1–6, 2016.

[54] Vicente Ortega y M.Ángeles Moya. Los orígenes del Sistema de Posicionamiento Global. http://catedraisdefe.etsit.upm.es/wiki/index.php/Los_or%C3%ADgenes_del_Sistema_de_Posicionamiento_Global_(GPS). Accessed: 13-03-2022.

[55] Kai Wang, Xing Yu, Qingyu Xiong, Qiwu Zhu, Wang Lu, Ya Huang, and Linyu Zhao. Learning to improve wlan indoor positioning accuracy based on dbscan-krf algorithm from rss fingerprint data. *IEEE Access*, PP:1–1, 05 2019.