Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA DE SEGOVIA

Grado en Ingeniería Informática de Servicios y Aplicaciones

Selección y optimización de algoritmos para la predicción de configuraciones en aeropuertos

---

Alumno: José Manuel Simón Ramos
Tutores: Aníbal Bregón Bregón
Jorge Silvestre Vilches

Agradecimientos

Quiero dar las gracias a mis tutores, D. Aníbal Bregón Bregón y D. Jorge Silvestre Vilches, y al profesor D. Miguel Ángel Martínez Prieto por darme la posibilidad de participar en este proyecto, así como por la ayuda y consejos prestados a lo largo de todo el desarrollo del mismo.

También dar las gracias a Boeing Research & Technology Europe (BR&T-Europe), y al profesor D. Pedro César Álvarez Esteban, por poner a disposición datos e información propia que han sido de gran importancia para la realización de este trabajo.

Finalmente agradecer a mi familia, amigos y a mi pareja por su apoyo a lo largo de todo este proyecto, ya que sin ellos no habría sido posible llegar hasta aquí.

Resumen

Con el paso del tiempo cada vez se ha vuelto más frecuente la utilización del transporte aéreo, ya sea para uso público, privado o comercial, y se espera que el número de vuelos sufra un incremento en los próximos años. Para poder hacer frente a ese volumen de tráfico aéreo, los aeropuertos han de disponer de una buena infraestructura, como por ejemplo número de pistas, orientación, ubicación de la terminal con respecto a las pistas o la configuración de estas. En especial, la configuración de las pistas (determinar que pistas se utilizan para despegar y cuales para aterrizar) es importante, ya que en ellas se llevan a cabo las maniobras de aterrizaje y de despegue. Existen muchos factores que pueden afectar a la configuración de las pistas como puede ser la velocidad y dirección del viento, las condiciones meteorológicas etc. Estas condiciones adversas pueden suponer que se tenga que realizar un cambio en la configuración del aeropuerto, suponiendo así un gran esfuerzo de recursos y retrasos en otros aterrizajes y despegues (entre otros problemas).

Es por ello que el ser capaces de predecir con una cierta antelación todos estos cambios futuros en la configuración que va a tener el aeropuerto (en las próximas una, dos o hasta tres horas), puede ser fundamental para planificar ese cambio con el tiempo suficiente para mantener la seguridad, evitar posibles accidentes y mejorar la eficiencia de las operaciones del aeropuerto (encajar mejor las salidas y llegadas, evitar posibles cambios de última hora, etc).

A partir de distintas fuentes de entrada (datos de vigilancia, información meteorológica, planes de vuelos completos, etc) procedentes del Aeropuerto de Barajas, se evaluarán y optimizarán distintos modelos de aprendizaje automático, con el objetivo de generar uno que permita resolver la problemática de la elección de la configuración de las pistas. Conjuntamente a esto se implementará un *dashboard* en el que se presente de una forma visual los resultados obtenidos a lo largo del análisis, así como permitiendo al usuario una interacción directa con el modelo de aprendizaje automático seleccionado.

Palabras clave: Configuración pistas, Aprendizaje Automático Supervisado, ADS-B, ATM.

Abstract

The use of air transport, whether for public, private or commercial purposes, has become increasingly common over time and the number of flights is expected to increase in the coming years. In order to be able to cope with this volume of air traffic, airports need to have a good infrastructure in place, such as number of runways, orientation, location of the terminal in relation to the runways or the configuration of the runways. In particular, the configuration of the runways (determining which runways are used for take-off and which for landing) is important, as it is here that the landing and take-off manoeuvres are carried out. There are many factors that can affect the configuration of the runway, such as wind speed and direction, weather conditions, etc. These adverse conditions can mean that a change in the airport configuration has to be made, thus involving a great effort of resources and delays in other landings and take-offs (among other problems).

Therefore, being able to predict all these future changes in the configuration that the airport will have (in the next one, two or even three hours), can be fundamental to plan ahead with enough time to fulfill the security regulations, avoid possible accidents and improve the efficiency of the airport operations (Improve the departures/arrivals distribution, avoid possible last minute changes, etc).

Using different sources of aeronautical data (surveillance data, meteorological information, complete flight plans, etc) coming from the Madrid-Barajas airport, a number of machine learning models will be evaluated and optimized, with the aim of generating one that allows to solve the problem of predicting the runway configuration. Together with this, a dashboard will be implemented to visually present the results obtained throughout the analysis, as well as allowing the user direct interaction with the selected machine learning model.

Keywords: Runway Configuration, Supervised Machine Learning, ADS-B, ATM.

Índice general

1. Introducción **1**
1.1. Motivación . . . 3
1.2. Objetivos . . . 4
1.3. Organización de la memoria . . . 4

2. Metodología de Trabajo **7**
2.1. Marco de Trabajo Scrum . . . 7
2.2. UVAGILE . . . 10
2.2.1. Herramientas utilizadas para la organización de UVAGILE . . . 12
2.3. Herramientas Utilizadas . . . 15
2.4. Tecnologías Utilizadas . . . 15

3. Estimación y Presupuestos **19**
3.1. Estimación . . . 19
3.2. Presupuestos . . . 27
3.3. Balance final del proyecto . . . 28
3.3.1. Planificación temporal . . . 28
3.3.2. Planificación económica . . . 30

4. Estudio del dominio **33**
4.1. Gestión del tráfico aéreo . . . 33
4.2. SESAR . . . 34
4.3. NextGen . . . 34
4.4. Sistemas de control ADS . . . 35
4.5. Pistas . . . 38
4.6. Configuración de las pistas . . . 41
4.7. Capacidad operativa . . . 43
4.8. Meteorología . . . 44
4.8.1. Meteorología en el aeropuerto . . . 46
4.9. Aeropuerto Adolfo Suárez Madrid-Barajas . . . 47

5. Aprendizaje Automático **49**
5.1. Estado del Arte . . . 49
5.2. Aprendizaje Automático . . . 50
5.3. Aprendizaje Supervisado . . . 52
5.3.1. K-Vecinos Más Cercanos . . . 54

5.3.2. Regresión Logística . . . 55
5.3.3. Árboles de Decisión . . . 57
5.3.4. Ensembles . . . 60
5.3.5. Redes Neuronales . . . 62

6. Modelo de Datos **69**
6.1. Raw Data . . . 69
6.1.1. Conjunto de datos inicial . . . 69
6.2. Generación del conjunto de datos final . . . 71
6.2.1. Obtención de la configuración a partir de utilización de las pistas . . . 73
6.3. Conjunto de datos final . . . 77
6.4. Extracción de información del modelo de datos . . . 77

7. Construcción del modelo **83**
7.1. Elección inicial de modelos . . . 83
7.1.1. Bosques aleatorios . . . 84
7.1.2. K-Vecinos Más Cercanos . . . 85
7.1.3. Regresión Logística . . . 87
7.1.4. Redes Neuronales . . . 88
7.2. Mejorando los modelos iniciales . . . 91
7.2.1. Bosques aleatorios . . . 92
7.2.2. K-Vecinos Más Cercanos . . . 95
7.2.3. Regresión Logística . . . 98
7.2.4. Redes Neuronales . . . 99
7.2.5. Comparativa . . . 103

8. Evaluación y Resultados **105**
8.1. Evaluación de los modelos . . . 106
8.2. Resultados . . . 110

9. Herramienta de Visualización **113**
9.1. Descripción de los actores . . . 113
9.2. Requisitos de Usuario . . . 114
9.3. Casos de Uso . . . 114
9.3.1. Especificación de los casos de uso . . . 115
9.4. Requisitos Funcionales . . . 117
9.5. Diseño . . . 118
9.5.1. Arquitectura Lógica . . . 118
9.5.2. Diseño de la aplicación . . . 119
9.6. Implementación de la herramienta . . . 123
9.7. Pruebas de Caja Negra . . . 125

10. Conclusiones y Trabajo Futuro **127**
10.1. Conclusiones . . . 127
10.2. Trabajo Futuro . . . 128
10.3. Aprendizaje personal . . . 129

Bibliografía **129**

Apéndices **135**

A. Contenido Adjunto **137**

B. Instalación y Versiones **141**
B.1. Versiones de las herramientas utilizadas 141
B.2. Instalación de la herramienta 141

C. Manual de Usuario **143**

Glosario **146**

Acrónimos **151**

Índice de figuras

1.1. Esquema de funcionamiento de ADS-B. Fuente: *https://www.airelectro.com/blog/ (visitado 16-03-2020).* 2

2.1. Ciclo de vida de un proyecto Scrum. Fuente: *Herizont.com (visitado 25-02-2020).* 9
2.2. Canal de Slack del proyecto. 13
2.3. Tablero del proyecto. 14

3.1. Diagrama de Gantt del proyecto. 26
3.2. Diagrama de Gantt final del proyecto. 32

4.1. Esquema objetivos del proyecto SESAR Fuente: *https://www.sesarju.eu/vision (visitado 14-03-2020).* 35
4.2. Estructura NextGen Fuente: *https://www.faa.gov/nextgen/what_is_nextgen/ (visitado 12-03-2020).* 36
4.3. Funcionamiento sistema ADS-B Fuente: *http://www.ads-b.com/ (visitado 14-03-2020).* 37
4.4. Representación utilizando la técnica de la rosa de los vientos, Fuente: *Wechoosethemoon (visitado 16-03-2020).* 39
4.5. Ejemplo de señalización en una pista de aeropuerto, Fuente: *Diario del Viajero (visitado 15-03-2020).* 41
4.6. Ejemplo configuración de pistas en un aeropuerto. 42
4.7. Factores meteorológicos que afectan a las operaciones del aeropuerto. Fuente: [17]. 45
4.8. Nube cumulonimbus, Fuente: *The Weather Channel (visitado 17-03-2020).* 46
4.9. Estados pista aterrizaje. 47
4.10. Distribución de pistas y configuraciones del Aeropuerto de Barajas. 48

5.1. Ejemplo clasificación nueva instancia para $k = 5$. Fuente: [24]. 54
5.2. Ejemplo de recta de regresión simple e hiperplano de regresión múltiple, Fuente: [24]. 56
5.3. Ejemplo árbol de decisión. 58
5.4. Ejemplo Votación. 61
5.5. Ejemplo *Bagging*. 61
5.6. Ejemplo *Boosting*. 62
5.7. Ejemplos Funciones Activación. Fuente: *DataCamp (visitado 31-03-2020).* 64
5.8. Ejemplos de conjuntos linealmente separables y no linealmente separables. Fuente: [24]. 65
5.9. Esquema del entrenamiento de una neurona artificial. Fuente: [24]. 65

5.10. Ejemplo algoritmo descenso por gradiente. 67

6.1. Configuración de las pistas para los meses de Enero, Febrero, Marzo y Abril (en azul: despegue; en naranja: aterrizaje). 74
6.2. Configuración en intervalos de 30 minutos (en azul: despegue; en naranja: aterrizaje). 75
6.3. Detección de configuraciones conflictivas en función del umbral de sensibilidad. . 76
6.4. Diagrama de clases. 78
6.5. Matriz de Correlación para todos los atributos de la aeronave y el vuelo. 79
6.6. Matriz de Correlación para los atributos a la hora de realizar la maniobra. 80
6.7. Matriz de Correlación para los atributos en función de factores meteorológicos. . 81

7.1. Precisión del modelo bosques aleatorios en función del número de estimadores. . . 85
7.2. Precisión del modelo K-Vecinos Más Cercanos en función del número de vecinos. 87
7.3. Precisión del modelo de Regresión Logística en función del algoritmo de resolución. 88
7.4. Precisión del modelo de Redes Neuronales en función de la activación. 89
7.5. Precisión global de los modelos de aprendizaje automáticos analizados. 91
7.6. Precisión Bosque Aleatorio (200 árboles) en función de la profundidad del árbol. 93
7.7. Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir una característica. 94
7.8. Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir un nodo hoja. 95
7.9. Precisión K-Vecinos Más Cercanos (5 vecinos) en función de la asignación de pesos. 96
7.10. Precisión K-Vecinos Más Cercanos (5 vecinos) en función del algoritmo para obtener la distancia entre los vecinos. 97
7.11. Precisión Regresión Logística en función de la penalización, el algoritmo de resolución y el parámetro C. 98
7.12. Precisión Redes Neuronales en función del número de capas y neuronas por capa. 100
7.13. Precisión Redes Neuronales para 4 capas ocultas (el punto verde representa el mayor valor de la precisión obtenida). 100
7.14. Precisión Redes Neuronales en función del factor de aprendizaje. 102
7.15. Curva de Validación Red Neuronal. 103

8.1. Ejemplo Validacion Cruzada. 105
8.2. Evaluación por el método de validación cruzada del modelo: **Bosques Aleatorios**. 106
8.3. Evaluación por el método de validación cruzada del modelo: **K-Vecinos Más Cercanos**. 107
8.4. Evaluación por el método de validación cruzada del modelo: **Redes Neuronales**. 108
8.5. Evaluación por el método de validación cruzada del modelo: **Clasificación por Votación**. 109
8.6. Comparativa global del método de validación cruzada para los modelos anteriores. 109
8.7. Matriz de Confusión y Métricas del modelo: **Bosques Aleatorios**. 110
8.8. Matriz de Confusión y Métricas del modelo: **K-Vecinos Más Cercanos**. 111
8.9. Matriz de Confusión y Métricas del modelo: **Redes Neuronales**. 111
8.10. Matriz de Confusión y Métricas del modelo: **Clasificación por Votación**. . . . 112

9.1. Diagrama de Casos de Uso de la aplicación. 115
9.2. Arquitectura Lógica de la aplicación. 119

A.1. Estructura de subdirectorios del directorio: */app*. 138
A.2. Estructura de subdirectorios del directorio: */scripts*. 139

B.1. Salida tras la ejecución del *dashboard*. 142

C.1. *Dashboard*: Pestaña de datos. 143
C.2. *Dashboard*: Pestaña de métricas. 144
C.3. *Dashboard*: Pestaña de evaluación. 145

Índice de tablas

3.1. Desglose de las tareas asociadas a la Historia 1 del Proyecto. 21
3.2. Desglose de las tareas asociadas a la Historia 2 del Proyecto. 22
3.3. Desglose de las tareas asociadas a la Historia 3 del Proyecto. 22
3.4. Desglose de las tareas asociadas a la Historia 4 del Proyecto. 23
3.5. Desglose de las tareas asociadas a la Historia 5 del Proyecto. 23
3.6. Desglose de las tareas asociadas a la Historia 6 del Proyecto. 24
3.7. Distribución de tareas por *sprints*. 25
3.8. Costes asociados a herramientas y recursos hardware. 27
3.9. Costes asociados a recursos humanos. Fuente: *Linkedin Salary (visitado 20-04-2020)*. 28
3.10. Distribución actualizada de tareas por *sprints*. 29
3.11. Desglose actualizado de las tareas asociadas a la Historia 2 del Proyecto. 30
3.12. Costes actualizados asociados a herramientas y recursos hardware. 31

4.1. Referencia para la asignación del código de referencia aeroportuario, Fuente: [29], [20]. 39
4.2. Anchura mínima de la pista para cada tipo de aeronave. Fuente: [20]. 40
4.3. Relación entre el número de franjas y la anchura de la pista. Fuente: *sociedadaeronautica.org (visitado 17-03-2020)*. 41
4.4. Capacidad Operativa para cada configuración de pista en función de las condiciones visuales. 44

5.1. Ejemplo Matriz de Confusión. 53

6.1. Descripción de las características del modelo de datos inicial. 70
6.2. Análisis características pertenecientes al vuelo. 70
6.3. Análisis características pertenecientes a la operación. 71
6.4. Análisis características pertenecientes a la meteorología. 71
6.5. Ejemplo codificación mediante etiquetas de clase y codificación en caliente. 72
6.6. Impacto de los intervalos conflictivos en los vuelos en función de la sensibilidad. . 75
6.7. Modificaciones realizadas para llevar a cabo la generación del modelo de datos final. 77

7.1. Precisión del modelo Bosques Aleatorios para cada distribución de datos. 85
7.2. Precisión del modelo K-Vecinos Más Cercanos para cada distribución de datos. . 86
7.3. Precisión del modelo de Regresión Logística para cada distribución de datos. . . . 87
7.4. Precisión del modelo de Redes Neuronales para cada distribución de datos en función de la activación. 89

7.5. Evolución precisión Redes Neuronales en función de la distribución de datos, el número de épocas y la función de activación. 90
7.6. Configuración base óptima de los modelos a mejorar. 91
7.7. Precisión Bosque Aleatorio (200 árboles) en función de la profundidad del árbol. 92
7.8. Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir una característica. 93
7.9. Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir un nodo hoja. 94
7.10. Modificaciones realizadas en los hiperparámetros del bosque aleatorio. 95
7.11. Precisión K-Vecinos Más Cercanos (5 vecinos) en función de la asignación de pesos. 96
7.12. Precisión K-Vecinos Más Cercanos (5 vecinos) en función del algoritmo para obtener la distancia entre los vecinos. 97
7.13. Modificaciones realizadas en los hiperparámetros del K-Vecinos Más Cercanos. . . 97
7.14. Precisión Regresión Logística para el penalizador L2 y el algoritmo de resolución Newton-cg en función del parámetro C. 99
7.15. Precisión Redes Neuronales en función del factor de aprendizaje y su modificación. 102
7.16. Modificaciones realizadas en los hiperparámetros de las redes neuronales. 103
7.17. Comparativa modelo base vs modelo con modificaciones. 104

9.1. Descripción del Actor-01: Usuario. 113
9.2. Requisitos de Usuario del sistema. 114
9.3. Casos de Uso del sistema. 114
9.4. Especificación CU-01: Listar vuelos. 116
9.5. Especificación CU-05: Predecir configuración de las pistas. 116
9.6. Requisitos Funcionales asociados al Caso de Uso CU-01: Listado de vuelos registrados en el sistema. 117
9.7. Requisitos Funcionales asociados al Caso de Uso CU-02: Filtrado de los datos de los vuelos registrados en el sistema. 117
9.8. Requisitos Funcionales asociados al Caso de Uso CU-03: Listado de las configuraciones de pistas del Aeropuerto de Barajas. 117
9.9. Requisitos Funcionales asociados al Caso de Uso CU-04: Listado de métricas de evaluación de los distintos modelos de aprendizaje automático. 118
9.10. Requisitos Funcionales asociados al Caso de Uso CU-05: Realizar predicción de la configuración de las pistas. 118
9.11. Requisitos Funcionales asociados a los elementos gráficos del sistema. 118
9.12. Diseño Interfaz DI-01: Mostrar datos de vuelo. 120
9.13. Diseño Interfaz DI-02: Mostrar métricas modelos. 121
9.14. Diseño Interfaz DI-03: Interacción con el modelo de aprendizaje automático. . . . 122
9.15. Prueba de Caja Negra CN-01: Comprobación existencia de todos los datos necesarios para predecir la configuración para un nuevo vuelo. 125
9.16. Prueba de Caja Negra CN-02: Existencia de los valores de una métrica para un modelo. 126
9.17. Prueba de Caja Negra CN-03: Validación del Callsign a la hora de predecir un vuelo. 126
9.18. Prueba de Caja Negra CN-04: Filtrado de los datos de vuelo en función de la configuración asignada. 126

B.1. Versiones de las herramientas y librerías utilizadas. 141

Capítulo 1

Introducción

En los últimos años, cada vez es más frecuente la utilización del transporte aéreo, ya sea para uso público, privado o comercial. Según el informe "Aviation: Benefits Beyond Borders" [11] elaborado por el *Air Transport Action Group* (ATAG), en octubre de 2018, la media de vuelos diaria registrada fue de 120.000 vuelos en los que viajaban en total 12 millones de personas aproximadamente, y cuya cifra se espera que aumente en los próximos años [3].

Para poder albergar esta cantidad de vuelos cada día, los aeropuertos deben disponer de una buena infraestructura (número y orientación de pistas, accesos, servicios en las terminales, ubicación de las terminales con respecto de las pistas, etc) y de una gestión eficaz y eficiente de las mismas. Es por ello por lo que la importancia de una buena Gestión del Tráfico Aéreo *(Air Traffic Management,* ATM, en inglés), cobra cada vez más fuerza con el paso de los años.

Para lograr los objetivos de mejorar la eficacia y eficiencia en las operaciones relacionadas con el tráfico aéreo, así como incrementar la seguridad de las mismas, han surgido distintas organizaciones e iniciativas internacionales que abordan estos objetivos y tratan de aportar nuevas soluciones en materia de coordinación, gestión operativa y regulación del espacio aéreo. Dentro del marco aéreo europeo, se encuentra Eurocontrol[1] *(European Organisation for the Safety of Air Navigation, Organización Europea para la Seguridad de la Navegación Aérea)*, una organización intergubernamental civil y militar creada en 1963, cuyo objetivo principal es la unificación del espacio aéreo europeo, de tal forma que se consiga agrupar e integrar en todos los países miembros, los servicios de navegación aérea: Servicios de Control del Tráfico Aéreo *(ATC)*, Servicios de Información de Vuelo *(FIS)*, servicios de alerta, etc.

Para cumplir con este objetivo, la Comisión Europea puso en marcha en 2004 el proyecto SES[2] *(Single European* Sky , Cielo Único Europeo), tratando así de eliminar la fragmentación que existe dentro del espacio aéreo, aumentar la eficiencia general del sistema, incrementar la seguridad, reducir los costes, disminuir los retrasos en los vuelos y aminorar el impacto ambiental [12]. A pesar de que esta iniciativa se lanzó hace 16 años, no se ha conseguido cumplir con esos objetivos hasta hace apenas un año [2].

Existen también otras iniciativas a nivel nacional como ENAIRE, el gestor de navegación aérea en España, cuyos objetivos son similares a los del proyecto SES: aumentar la seguridad, reducir los costes y disminuir el impacto medioambiental. Uno de los proyectos de esta iniciativa es el proyecto SACTA *(Sistema Automatizado de Control de Tránsito Aéreo*[3]*)* destinado a facilitar

[1] https://www.eurocontrol.int/.

[2] https://ec.europa.eu/transport/modes/air/ses_en.

[3] https://www.enaire.es/servicios/atm/sistemas_de_gestion_del_transito_aereo_atm/sacta.

la prestación de los servicios de tráfico aéreo de los que es responsable ENAIRE.

Por otra parte, fuera del marco aéreo europeo, se encuentra el proyecto NextGen *(The Next Generation Air Transportation System)*, un proyecto enfocado al transporte aéreo de Estados Unidos cuya misión es aumentar la seguridad en los vuelos, mejorar la eficiencia y hacer más predecible su sistema de gestión, coordinando los servicios de navegación que disponen actualmente [1].

En términos generales, el avance de la tecnología ha facilitado este proceso de gestión de forma coordinada. En particular, hay que destacar ADS-B *(Automatic Dependent Surveillance Broadcasting)*, una tecnología de vigilancia y monitorización que pretende sustituir a los sistemas de radares convencionales, combinando el posicionamiento por satélite con un enlace de datos por radiofrecuencia. Con esto se consigue que la aeronave envíe de forma continua, periódica y automática los datos obtenidos mediante sus sensores (posicionamiento, altitud, velocidad, etc) a todos aquellos receptores que se encuentren dentro de su rango de influencia (controladores de vuelo, estaciones de tierra u otras aeronaves que admitan este tipo de tecnología), en lugar de limitarse a responder únicamente las peticiones de información procedentes de las estaciones de tierra.

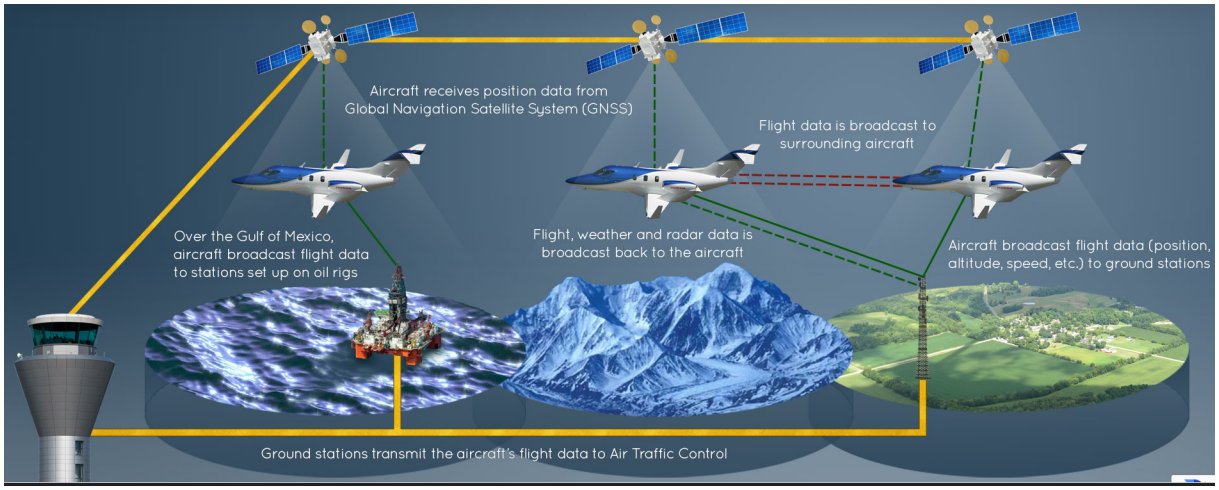


Figura 1.1: Esquema de funcionamiento de ADS-B. Fuente: *https://www.airelectro.com/blog/ (visitado 16-03-2020).*

Uno de los factores que más influyen en la gestión del tráfico aéreo es la configuración de las pistas durante las operaciones de aterrizaje y despegue. Esta configuración es de suma importancia a la hora de mejorar la eficiencia del tráfico aéreo, ya que el sistema de configuración de las pistas es el principal cuello de botella en la capacidad del aeropuerto [14]. Esto se debe a que el aeropuerto es el único punto en el que coinciden los aviones durante el vuelo, y en el que tanto las maniobras de aterrizaje como las de despegue no se efectúan hasta que la realización de las mismas no sea totalmente segura. Por tanto, una mala configuración en las pistas puede llegar a suponer un retraso en los tiempos de salida y llegada debido a una acumulación de aviones esperando para efectuar dichas maniobras, afectando así a la capacidad del aeropuerto y a la red de transporte aérea.

Otro beneficio de establecer una correcta gestión en la configuración de las pistas es la reducción en los tiempos de espera, ya que las modificaciones en la configuración de las pistas no

se realizan de forma instantánea, y un cambio en la configuración de las mismas tarda aproximadamente 30 minutos en hacerse efectivo de forma segura. Durante este periodo de tiempo, la capacidad operativa de las pistas afectadas disminuye debido al cambio en la configuración, haciendo que el aeropuerto pierda temporalmente parte de su infraestructura, suponiendo un aumento en el tiempo de espera de las aeronaves que necesiten utilizar las pistas afectadas.

Es importante destacar que, además de reducir los tiempos de espera y aumentar la fluidez en el tráfico aéreo, la seguridad de los pasajeros y del personal de a bordo en los aviones es otro de los factores que se ve mejorado. El motivo de esto es que, como mencionaremos más adelante, la asignación de una pista a una aeronave se encuentra condicionada, entre otras cosas, a la climatología en el momento de realizar las operaciones de aterrizaje o despegue, procurando en todo momento que estas maniobras se realicen en la pista disponible en la que el impacto meteorológico en la aeronave sea lo menor posible. De esta manera, se persigue minimizar el riesgo inherente a estas operaciones, de manera que se lleven a cabo en las condiciones más seguras posibles.

Por tanto, podemos concluir con que la gestión de la configuración de las pistas es una tarea muy importante dentro de la gestión del tráfico aéreo. Una buena configuración de las pistas logra aumentar la seguridad de los pasajeros, reduce los tiempos de espera para efectuar las maniobras de despegue y aterrizaje, disminuye los costes que genera una mala gestión del tráfico y mejora la eficiencia y rendimiento de las infraestructuras del aeropuerto.

1.1. Motivación

Uno de los factores determinantes a la hora de mejorar la gestión del tráfico aéreo y reducir tanto los tiempos de espera como los retrasos, es la elección de la configuración de las pistas. La elección de la configuración ha de establecerse de tal forma que se consiga favorecer la realización de las maniobras de aterrizaje y de despegue con unas condiciones lo más favorables posibles, además de minimizar el número de veces que se realizan estas modificaciones. La selección eficiente en esta configuración permite reducir el coste de mantenimiento de las pistas, así como aumentar la capacidad operativa del aeropuerto.

Este trabajo tiene como propósito ayudar a resolver la problemática de la elección de pistas, generando un modelo de aprendizaje automático que proporcione apoyo al controlador aéreo a la hora de determinar qué configuración de pistas elegir. Actualmente, la elección de la configuración es una decisión llevada a cabo por parte de los controladores aéreos que se realiza en función tanto de las condiciones que se den en un determinado momento cómo de las condiciones previstas a corto plazo. Debido a la gran cantidad de factores que afectan a esta decisión y a la gran complejidad de la misma, es más probable que puedan ocurrir fallos debido al error humano, por lo que este sistema pretende minimizar ese error y mejorar la eficacia y eficiencia a la hora de determinar qué configuración de pistas elegir.

Para ello se llevará a cabo el análisis, estudio y prueba de distintos algoritmos de aprendizaje automático que puedan ser efectivos para esta problemática, seleccionando y refinando aquellos que realicen una mejor predicción sobre la elección de la configuración en las pistas.

1.2. Objetivos

El presente Trabajo Fin de Grado afronta la problemática de predecir la configuración de las pistas en los aeropuertos utilizando los datos obtenidos a partir de los mensajes ADS-B provenientes de las aeronaves próximas al aeropuerto junto con los datos meteorológicos de la zona. En esta línea de trabajo se identifican los siguientes objetivos a abordar:

OBJ-1 Procesamiento y Análisis de los datos de los vuelos.

- OBJ-1.1 Realización de una análisis exploratorio inicial de los datos.
- OBJ-1.2 Definición del modelo de datos final.
- OBJ-1.3 Transformación y normalización de los datos de acuerdo al modelo de datos final.
- OBJ-1.4 Elaboración de un análisis de las dependencias e influencias existente entre las distintas características de los datos.

OBJ-2 Prueba y refinamiento de modelos de aprendizaje automático.

- OBJ-2.1 Estudio de algoritmos de aprendizaje automático que pueden utilizarse para la resolución del problema.
- OBJ-2.2 Realización de una comparativa entre los distintos algoritmos con su configuración inicial y evaluar su desempeño.
- OBJ-2.3 Refinamiento de los modelos de aprendizaje automático realizando ajustes sobre sus hiperparámetros para mejorar su desempeño final.

OBJ-3 Evaluación y selección del modelo de aprendizaje automático final.

- OBJ-3.1 Aplicación de técnicas de evaluación para comprobar el desempeño real de los modelos de aprendizaje automático refinados.
- OBJ-3.2 Análisis de los modelos refinados y selección del modelo final.

OBJ-4 Diseño e implementación de un *dashboard* de visualización e interacción con el modelo.

Con la consecución de los objetivos mencionados anteriormente se pretende obtener un modelo de aprendizaje automático sobre el que el controlador pueda apoyarse para poder determinar la configuración de las pistas de un aeropuerto, aumentando tanto la eficiencia del aeropuerto como la seguridad de las aeronaves y disminuyendo los costes asociados al mantenimiento de las pistas.

1.3. Organización de la memoria

El presente documento se estructura en capítulos en los que se tratan diferentes aspectos del proyecto. Los capítulos que conforman el documento y su temática son:

Capítulo 1. Introducción: A lo largo del Capítulo 1 se realizará una introducción sobre el contexto en el que se desarrolla este proyecto, así como la motivación del mismo y los objetivos que se pretenden conseguir con su finalización.

Capítulo 2. Metodología de Trabajo: El Capítulo 2 describe la metodología utilizada para llevar a cabo el desarrollo del proyecto así como las herramientas y tecnologías que han sido empleadas tanto para la implementación, como para la gestión y control de todos los aspectos relativos al mismo.

Capítulo 3. Estimación y Presupuestos: En el Capítulo 3 se presenta la planificación temporal que se llevará a cabo a lo largo del trabajo, en la que se incluirán todas aquellas tareas que se han derivado a partir de los objetivos del proyecto. Finalmente, el capítulo concluye con el desglose de los costes asociados al proyecto basados en la planificación temporal llevada a cabo.

Capítulo 4. Estudio del Dominio: A lo largo del Capítulo 4 se realiza un recorrido del ecosistema de la aviación, en el que engloban las tareas de gestión del tráfico aéreo que afectan a la hora de establecer las configuraciones de las pistas, así como las distintas propuestas que se han creado con la finalidad de mejorar y optimizar esta gestión. Además de esto, en este capítulo se trata también la casuística asociada a la elección de la configuración de las pistas, tanto desde el punto de vista del aeropuerto, como desde el punto de vista de la meteorología.

Capítulo 5. Aprendizaje Automático: En el Capítulo 5 se presenta el concepto de aprendizaje automático, así como los distintos tipos de aprendizaje automático que existen. Además, se describen los algoritmos empleados tanto en el presente proyecto, como en propuestas similares realizadas anteriormente.

Capítulo 6. Modelo de Datos: En el Capítulo 6 se realiza la descripción del proceso llevado a cabo para transformar el conjunto de datos original (*raw data*) en el conjunto de datos final, así como la obtención de la configuración de las pistas a partir de estos datos, para posteriormente, llevar a cabo el entrenamiento, prueba y evaluación de los distintos modelos de aprendizaje automático.

Capítulo 7. Construcción del modelo: En el Capítulo 7 se hace un recorrido por los distintos modelos de aprendizaje automático que se han utilizado para resolver el problema que ocupa el proyecto, así como las distintas modificaciones que se han aplicado a los mismos para optimizarlos y mejorar los resultados que ofrecen.

Capítulo 8. Evaluación y Resultados: En el Capítulo 8 se lleva a cabo un estudio de los modelos generados en el Capítulo 7 en el que se analiza el comportamiento de estos tras el proceso de refinamiento en el que se analiza su efectividad para la determinación de configuraciones de pistas.

Capítulo 9. Herramienta de Visualización: En el Capítulo 9 se lleva a cabo un análisis software de la herramienta que se va a desarrollar: actores del sistema, requisitos de usuario, casos de uso, diseño etc.

Capítulo 10. Conclusiones y Trabajo Futuro: En el Capítulo 10 se enuncian las conclusiones obtenidas a lo largo del desarrollo del proyecto y se presentan posibles líneas de trabajo que podrían continuarse en el futuro a partir de los resultados obtenidos.

Apéndices: En este apartado del documento se encuentra información adicional asociada a la entrega del proyecto, así como manuales y detalles de la herramienta desarrollada. Este apéndice se encuentra constituido por las siguientes secciones:

- **Apéndice A. Contenido Adjunto:** En esta sección del apéndice se muestra el contenido adjunto a la entrega de este proyecto, así como la explicación del mismo y su distribución.
- **Apéndice B. Instalación y Versiones:** En esta sección del apéndice se explica el proceso a seguir para llevar a cabo la instalación de la herramienta así como las versiones del software y librerías utilizadas para su desarrollo.
- **Apéndice C. Manual de usuario:** En esta sección del apéndice se realiza una explicación de la cada una de las funcionalidades que dispone de la herramienta.

Glosario: En este apartado se encuentran definidos conceptos, terminología y acrónimos importantes que han ido surgiendo a lo largo del desarrollo del presente documento.

Capítulo 2

Metodología de Trabajo

La realización del presente Trabajo Fin de Grado se va a realizar de forma iterativa e incremental de acuerdo a los principios establecidos en el Marco de Trabajo *Scrum* adaptados al ámbito académico: UVAGILE.

2.1. Marco de Trabajo Scrum

Scrum es un Marco de Trabajo a través del cual las personas pueden abordar problemas complejos adaptativos a la vez que se entregan productos de forma eficiente y creativa con el máximo valor [28]. Este Marco de Trabajo se centra en mejorar la inspección, la adaptación y la transparencia del proyecto, haciendo uso de sus valores principales: el coraje (los miembros del equipo *Scrum* tienen el coraje para hacer siempre lo correcto y trabajar para resolver juntos los problemas), la focalización (las personas involucradas se centran en cumplir las tareas del *sprint* y sus objetivos), el compromiso (todos los miembros están comprometidos para conseguir el triunfo del equipo), el respeto (los miembros del equipo siempre se respetan los unos a los otros) y la apertura (todos los involucrados comparten información con el objetivo de mejorar el resultado del producto final).

Para cumplir con estos objetivos, *Scrum* se define en 3 roles (el conjunto de todos ellos se define como *Scrum Team*), 5 eventos y 3 artefactos (ver Figura 2.1).

Los roles que conforman la metodología *Scrum* son:

- **Scrum Master:** El *Scrum Master* es el líder del equipo de trabajo ágil. Su misión consiste en hacer que el equipo cumpla con los objetivos marcados hasta alcanzar la finalización del último *sprint* del proyecto. Para ello, se encarga de solucionar cualquier problema que aparezca durante el avance del proyecto. Por otra parte, el *Scrum Master* es el encargado de que se realicen correctamente todas las ceremonias que forman parte del Marco de Trabajo *Scrum*, dicho de otro modo, es el responsable de que se sigan y se cumplan las prácticas y valores del Marco de Trabajo *Scrum*.

- **Product Owner:** Es el nexo entre las personas interesadas fuera de la organización (*stakeholders*) y el equipo de desarrollo interno. Al ser ese nexo, la figura de *Product Owner* se encuentra a un nivel diferente al del *Scrum Master*, ya que el *Scrum Master* no dispone de un contacto directo con el cliente. Además de esto, el *Product Owner* es el responsable de maximizar el valor del producto entregado, ya que, al encontrarse en contacto directo

con los *stakeholders*, conoce de primera mano qué características del producto aportan un mayor valor, para posteriormente, transmitir esta información al equipo de desarrollo y que estos definan y prioricen las tareas necesarias para implementar dichas características. Finalmente, el *Product Owner* es el encargado de organizar, modificar y gestionar el *Product Backlog* con la información que obtiene a través de las reuniones con los *stakeholders*.

- **Equipo de Desarrollo:** El Equipo de Desarrollo se encuentra constituido por todas aquellas personas que se encargan del desarrollo del producto. Su objetivo dentro del proyecto es el de derivar las tareas necesarias para satisfacer los criterios de aceptación asociados a cada historia, establecer la complejidad de las tareas derivadas a partir de las historias y negociar el alcance del *sprint* con el *Product Owner*.

Una de las características del Marco de Trabajo *Scrum* es la retroalimentación *(feedback)* que se genera debido a la inclusión del cliente en el desarrollo del proyecto, ofreciéndole un producto funcional al final de cada *sprint*, para así identificar aquellas acciones que se han hecho bien y en cuáles se ha de mejorar.

Para obtener esta retroalimentación, detectar, gestionar y corregir errores, y organizar el siguiente *sprint* de trabajo, el Marco de Trabajo *Scrum* dispone de los siguientes eventos:

- **Sprint:** El *sprint* es un espacio temporal (de tamaño máximo 1 mes), cuyo objetivo consiste en realizar la entrega de un incremento de producto funcional y utilizable. Estos *sprints*, han de ser todos de la misma duración, tener un objetivo fijado a cumplir tras la finalización de éste y disponer de la información necesaria que permita alcanzar dichos objetivos. El objetivo del *sprint* es invariable, por lo que durante el mismo no se realiza ninguna tarea que no ayude a lograr dicho objetivo. El *sprint* se encuentra constituido por 4 eventos: 1) *Sprint Planning*; 2) *Daily Scrum*; 3) *Sprint Review* y 4) *Sprint Retospective*.

- **Sprint Planning:** El *Sprint Planning* es una reunión que se realiza al principio de cada *sprint*. Es una reunión extensa (puede durar un máximo de 8 horas) y tiene como finalidad la planificación del *sprint*, decidiendo qué tareas se van a abordar y cómo se van a realizar. En esta reunión participan el equipo de desarrollo, el *Scrum Master* y el *Product Owner*. Este último se encargará de establecer tanto el objetivo del *sprint* como las historias del *Product Backlog* que se han de completar, pero siempre, sujeto a la decisión final del equipo de desarrollo (puesto que es posible que el objetivo marcado por el *Product Owner* sea imposible de alcanzar durante el *sprint* debido a su complejidad o su tamaño). Tras la finalización de este evento se habrán obtenido respuestas a las siguientes preguntas:

  - ¿Cuál es el objetivo a que se va a cumplir en este *sprint*?
  - ¿Cómo se conseguirá cumplir dicho objetivo?

- **Daily Scrum:** La *Daily Scrum* es una reunión breve (10 - 15 minutos aproximadamente) en la que el equipo de desarrollo se "sincroniza" para planificar el trabajo de las próximas 24 horas. Por ello, a esta reunión únicamente asiste el equipo de desarrollo, aunque tanto el *Scrum Master* como el *Product Owner* pueden ser invitados a asistir. La presencia del *Scrum Master* en la *daily* se limita a facilitar la fluidez de la ceremonia, haciendo que no se sobrepasen los tiempos marcados y que la reunión se realice correctamente. Durante la ceremonia, cada uno de los miembros del equipo de desarrollo ha de actualizar el tablero

de trabajo con las tareas que haya realizado y exponer al resto del equipo la respuesta de las siguientes preguntas:

- ¿Qué hiciste la ayer?
- ¿Que harás hoy?
- ¿Existe algún impedimento que te impide avanzar?

- **Sprint Review:** La *Sprint Review* es una ceremonia de duración máxima 4 horas realizada a la finalización del *sprint*, en la que el equipo de desarrollo y el *Product Owner* muestran a los *stakeholders* el incremento funcional del producto, a la vez que estos colaboran para determinar qué cosas se pueden realizar que permitan aumentar el valor. Gracias a esto se consigue detectar con mayor rapidez aquellas cosas que se han de realizar o que se deban cambiar para disminuir la brecha entre lo desarrollado por el equipo de desarrollo y lo esperado por parte de los *stakeholders*. Finalmente, de este evento se consigue información relevante que posteriormente se utilizará para refinar las tareas asociadas a una determinada historia.
- **Sprint Retospective:** La *Sprint Retrospective* es una reunión de un máximo de 3 horas que se realiza al finalizar cada *sprint*. En esta ceremonia participa todo el *Scrum Team* (equipo de desarrollo, *Scrum Master* y *Product Owner*). La finalidad principal de este evento es la de analizar cómo ha sido el transcurso del *sprint*: si se han cumplido los objetivos propuestos, qué se ha aprendido durante el desarrollo de este, qué acciones de mejora se llevarán a cabo para el siguiente *sprint*, consolidar el plan de mejora, etc.

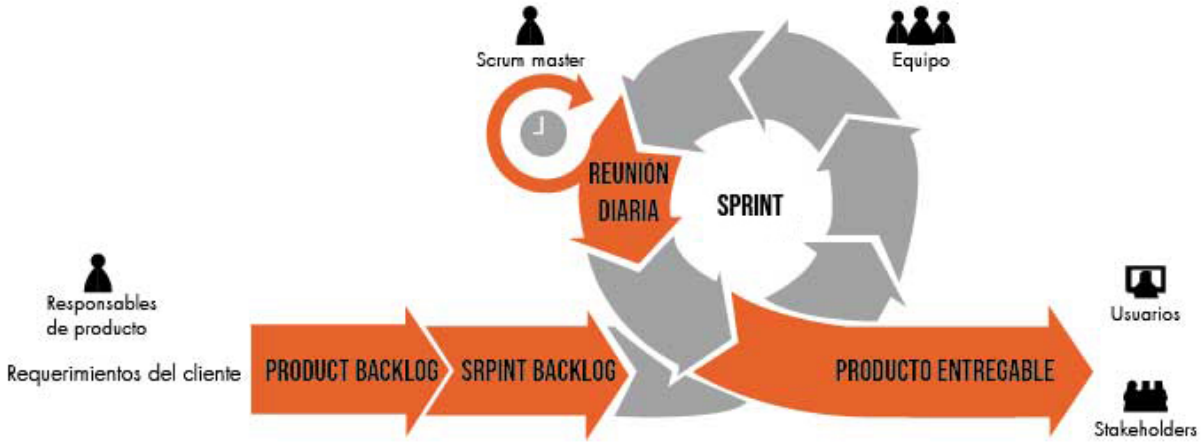


Figura 2.1: Ciclo de vida de un proyecto Scrum. Fuente: *Herizont.com (visitado 25-02-2020).*

Para organizar y gestionar cada uno de los *sprints* así como las tareas necesarias que permitan obtener el producto final a desarrollar, se utilizan los siguientes artefactos:

- **Product Backlog:** El *Product Backlog* es una lista ordenada de todas las características, funcionalidades, requisitos o mejoras que se han de realizar en el producto. Las tareas recogidas en el *Product Backlog* cuentan con una descripción en la que se indica la motivación

por la cual es necesaria esa característica en el sistema y lo que le aporta al mismo, así como una estimación del coste temporal que supondría realizar dicha tarea. Por otra parte, a partir de estas tareas, se generarán otras más pequeñas, cuya consecución supondrá la finalización de la tarea a la que pertenece. El encargado de organizar, gestionar y modificar este artefacto es el *Product Owner*.

- **Sprint Backlog:** El *Sprint Backlog* es una lista de tareas cuya consecución implica el cumplimiento de los objetivos fijados en el *sprint*, por lo que, **únicamente** contiene elementos que permitan alcanzar los objetivos de este. En este artefacto se colocan aquellos *ítems* del *Product Backlog* que permitan maximizar el valor obtenido al finalizar el *sprint*, y que a su vez la duración de las mismas sea menor que la del *sprint*. A partir de estos *ítems* se generarán tareas que permitan abordar de una forma más eficaz la tarea de la que dependen. Los únicos responsables de gestionar este artefacto son los miembros del equipo de desarrollo.

- **Incremento:** El incremento consiste en la suma de todos aquellos elementos pertenecientes al *Product Backlog* que han sido completados tras la finalización del *sprint* (incluidos los de todos los *sprints* anteriores). Este incremento deberá cumplir todos los requisitos descritos en las tareas que lo conforman y deberá estar terminado al finalizar el srpint independientemente de que el *Product Owner* decida o no entregárselo a los *stakeholders*.

2.2. UVAGILE

UVAGILE [16] nace como un proyecto de innovación docente cuya finalidad principal reside en la aplicación de metodologías ágiles a la docencia universitaria. El objetivo principal de esta metodología es la implantación de los valores y principios ágiles en la enseñanza y el aprendizaje a nivel universitario, fomentar la comunicación entre profesores y alumnos, impulsar la organización por parte del alumno y promover la cooperación y coordinación entre ellos.

Debido a que este Marco de Trabajo se está utilizando actualmente en la docencia, y por tanto su utilización en este tipo de proyectos aún no se encuentra consolidada, se van a introducir algunas modificaciones sobre lo establecido inicialmente por la metodología para adaptarla a las particularidades de un Trabajo Fin de Grado.

El ciclo de vida que va a tener nuestro proyecto es el mismo que el que puede tener cualquier proyecto que utilice *Scrum*: Comenzaremos introduciendo en el ***Product Backlog*** todas aquellas tareas que conforman nuestro proyecto. A continuación, al comienzo de cada *sprint*, seleccionaremos aquellas tareas que van a ser abordadas en dicho *sprint* (todas estas tareas constituyen nuestro ***Sprint Backlog***). Una vez sabemos qué tareas son las que tenemos que realizar durante el *sprint* pasamos a la fase de desarrollo, realizando aquellas reuniones intermedias previstas en Scrum. Finalmente, una vez finalizado el *sprint*, obtenemos un **Producto Entregable** que describe una versión funcional del producto final. La finalidad de esto es poder mostrárselo al cliente para obtener *feedback* por su parte y poder realizar mejoras en sucesivos *sprints*. Una vez hemos terminado el *sprint*, comenzamos a preparar el siguiente y así sucesivamente hasta obtener el producto final esperado.

Para la implementación en nuestro proyecto, se ha acordado una duración de *sprints* de tres semanas, comenzando el primer *sprint* el día *05/02/2020* y el último el *10/06/2020*. Los roles que se van a adoptar junto con la figura que lo va a adoptar son los siguientes:

- **Scrum Master:** En este caso, al tener únicamente un miembro en el equipo de desarrollo (el alumno), será este el encargado de adoptar el rol de *Scrum Master* a lo largo del desarrollo del proyecto.
- **Product Owner:** La figura del *Product Owner* la adoptará el tutor del proyecto dado que será la persona encargada de fijar las tareas que hay que realizar y priorizará aquellas que se tengan que finalizar antes. Además de esto, será el nexo entre el equipo de desarrollo (alumno) y los *stakeholders* del proceso (los requisitos que debe cumplir un Trabajo Fin de Grado, el tribunal de la defensa, etc).
- **Equipo de Desarrollo:** Al igual que ocurría con el *Scrum Master*, el equipo de trabajo se encuentra constituido únicamente por el alumno, con lo cual será este el que adopte el rol del equipo de trabajo a lo largo de todo el proyecto.

Por otro lado, debido a que el desarrollo del proyecto va a ser realizado por una única persona y por tanto el ritmo de avance es menor, se van a realizar pequeñas variaciones en cuanto a las ceremonias que conforman *Scrum*, para así poder adaptarlas a esta situación. Por ello, las ceremonias que se llevarán a cabo a lo largo del proyecto son:

- **Inicio del Proyecto:** Se realizará al comienzo del proyecto y participarán tanto el *Scrum Master* como *Product Owner*. La finalidad de esta reunión consiste en establecer las historias de usuario iniciales del proyecto así como definir la planificación del primer *sprint* del proyecto.
- **Weekly:** La "Weekly" es una reunión breve (10 - 15 minutos aproximadamente) que se realiza cada semana y en la que participan tanto el *Scrum Master* como el *Product Owner*. Durante esta ceremonia se contestarán las siguientes preguntas:

  - ¿Qué hiciste la semana pasada?
  - ¿Que harás esta semana?
  - ¿Existe algún impedimento que te impide avanzar?

  Esta ceremonia es la equivalente a la *Daily* que se realiza en *Scrum* pero debido a que el desarrollo del proyecto lo lleva a cabo una única persona, el avance del proyecto no es lo suficientemente significativo como para que la realización de esta ceremonia de manera diaria sea necesaria y aporte valor. De la misma manera, en lugar de servir para sincronizar al equipo de desarrollo, su objetivo principal es que el alumno pueda comunicar los avances al tutor con regularidad.
- **RetroPlanning:** Se realiza al principio de cada *sprint*. Consiste en una reunión más extensa que la *Weekly* (1h aproximadamente). En esta ceremonia participan tanto el *Scrum Master* como el *Product Owner* y está dividida en dos partes:

  1) **Retrospectiva:** Durante esta fase de la ceremonia se comentará cómo ha sido el transcurso del *sprint*, si se ha conseguido completar el objetivo marcado para el mismo, así como las dificultades que han ido surgiendo a lo largo de este. Una vez hecho esto se obtendrán qué acciones llevadas a cabo han ayudado a conseguir los objetivos marcados, y cuáles de ellas se podrían mejorar, estableciendo un plan de mejora que

se aplicará en el siguiente *sprint* con el objetivo de mejorar aquellos aspectos que no se han realizado correctamente.

2) **Sprint Planning:** Esta parte de la ceremonia está enfocada en la planificación. Durante esta reunión se establecerán aquellas tareas o historias que se van a abordar durante el *sprint* actual, agregándolas al *Sprint Backlog*, así como la creación de nuevas historias y tareas en el *Product Backlog* que puedan ir surgiendo a medida que avanza el proyecto.

Esta es la ceremonia más importante de todas las que se van a realizar puesto que es el cierre de un *sprint* (por tanto se finalizarán y validarán las tareas que se han realizado durante el mismo) y el inicio de otro y por tanto se ha de realizar la planificación del nuevo *sprint*.

- **Reuniones "Socorro":** Se realizarán de manera excepcional durante el desarrollo del proyecto en el caso de que ocurra algún problema que no nos permita continuar con el avance del proyecto. En ellas participarán tanto el *Product Owner* como el *Scrum Master* y su duración no está definida puesto que dependerá del problema a tratar.

  Se solicitará la realización de estas reuniones cuando aparezca un problema durante el desarrollo del proyecto y que debido a su complejidad no pueda ser abordado en la *Weekly* y que por su impacto en el proyecto no pueda posponerse hasta el *RetroPlanning*.

Finalmente, para realizar esta implementación es aconsejable la utilización de un conjunto de herramientas que nos faciliten el trabajo. Por tanto las herramientas que seleccionemos han de cubrir los siguientes requisitos:

- Creación y uso de canales de comunicación entre el tutor y el alumno.
- Gestión del tablero del proyecto.
- Control de las fechas importantes del proyecto: ceremonias y comienzo-fin de un *sprint*.
- Almacenamiento de los archivos del proyecto de tal forma que se encuentren accesible en todo momento para todos los miembros implicados en el proyecto.

La selección de las herramientas que cumplan con estos requisitos y su utilización en el proyecto se encuentran especificadas en la Sección 2.2.1.

2.2.1. Herramientas utilizadas para la organización de UVagile

Como se ha mencionado anteriormente, el uso de este tipo de metodologías requiere una gran organización y autogestión por parte del alumno, siendo este el encargado de organizar sus tareas y de cumplirlas en los tiempos fijados. Además de esto, es necesaria la existencia de una buena comunicación entre el tutor del proyecto y el propio alumno, consiguiendo así que el tutor tenga constancia en todo momento del avance del proyecto y de los problemas que van surgiendo durante el desarrollo del mismo, de tal forma que estos se puedan resolver de una forma más rápida y eficaz.

Las herramientas utilizadas para la gestión y coordinación durante todo el proyecto son las siguientes:

- **Slack:** Es una aplicación de mensajería destinada a equipos de trabajo. Esta herramienta será la que se utilizará en todo momento para realizar la comunicación no presencial entre el tutor y el alumno (ver Figura 2.2).

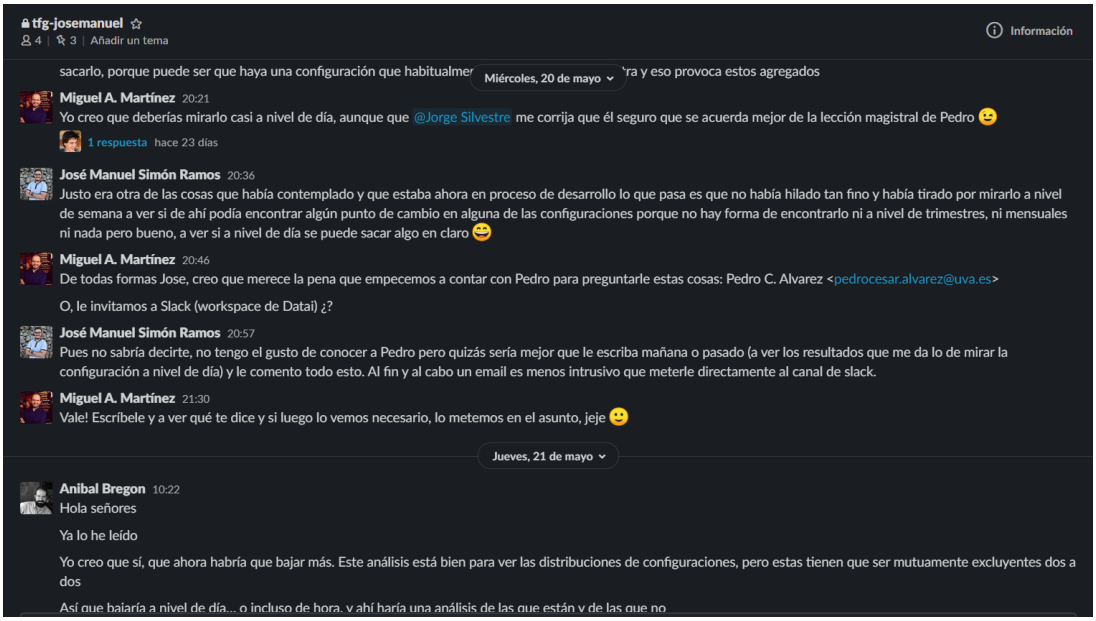


Figura 2.2: Canal de Slack del proyecto.

- **Trello:** Trello es una herramienta de administración de proyectos cuya estructura está basada en un **Tablero Kanban** (conjunto de columnas que representan listas en las cuales añadiremos diversas tarjetas que se corresponderán con las tareas a realizar).

  El uso que se le va a dar a Trello durante el desarrollo del proyecto es de organización y control de todo el avance del mismo, realizando una división en columnas donde se clasificarán las tareas que conforman el proyecto (ver Figura 2.3). Esta división se ha realizado de la siguiente forma:

  - **H_BACKLOG:** En esta columna se incluirán las historias en las que dividiremos el proyecto. Cada una de las historias representa un requisito o funcionalidad que vamos a añadir a nuestro sistema. Para abordar de una forma más sencilla estas historias, vamos a descomponerla en tareas más simples de manera que una historia se considerará completada cuando se terminen y validen todas las tareas que tenga asociadas.
  - **T_BACKLOG:** Conjunto de tareas más simples que se han obtenido a partir del desglose de otras más complejas pertenecientes a la columna *H_BACKLOG*.
  - **SPRINT_ACTUAL:** Todas aquellas tareas que serán abordadas durante el *sprint* en el que nos encontremos.
  - **T_HACIENDO:** Todas aquellas tareas que se están realizando actualmente.
  - **T_BLOQUEADAS:** Aquellas tareas pertenecientes al *sprint* en el que nos encontremos, cuyo avance haya sido nulo debido a algún problema (tecnológicos, desconocimiento de lo que se ha de hacer debido a una falta de especificación en dicha tarea, etc).

- **T_COMPLETADAS:** En esta columna se incluirán todas aquellas tareas que se hayan completado y que cumplan los criterios de aceptación asociados a la misma. A pesar de que la tarea se encuentre en esta columna, no se considerará completada hasta que no sea revisada y validada por el tutor del proyecto, siendo este el que dé el visto bueno a la misma.
- **H_COMPLETADAS:** En esta columna se incluirán aquellas historias de la columna *H_BACKLOG* cuyas tareas en las que ha sido dividida han sido completadas y validadas.

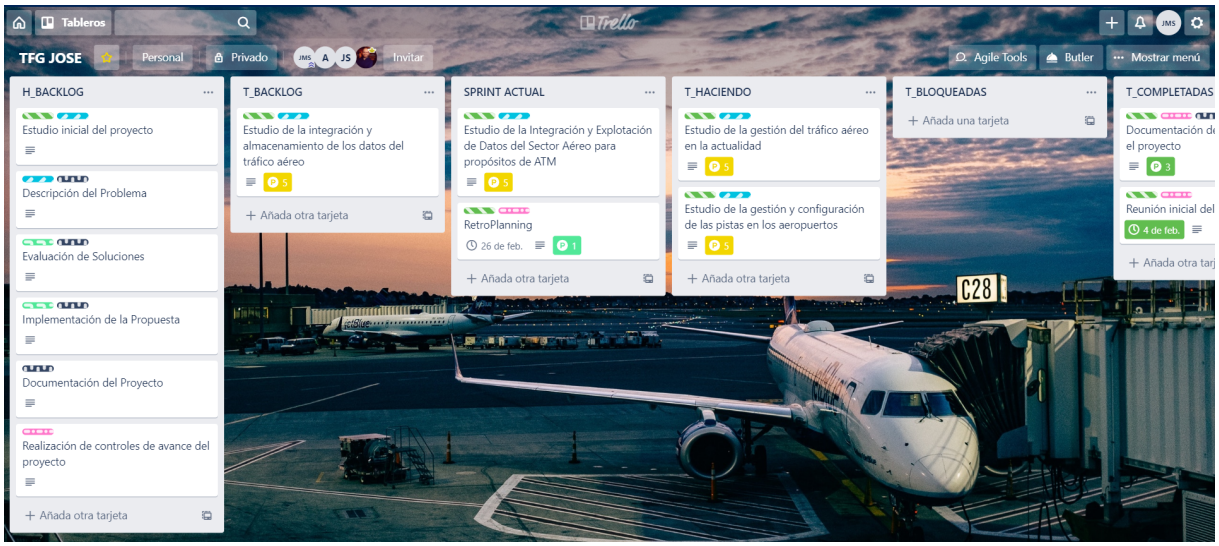


Figura 2.3: Tablero del proyecto.

Otro factor importante para la organización del proyecto es la asignación y control de los puntos de historia *(story points)* que asignamos a cada tarea. Un *story point* es la unidad de medida habitual en marcos de trabajo ágiles para representar el esfuerzo que supone completar una determinada tarea. Esta unidad no dispone de una equivalencia objetiva y estandarizada con el número de horas empleadas, ya que el "valor" de un *story point* varía en función de algunos factores, como la experiencia del equipo de trabajo, el avance del proyecto o el propio equipo de desarrollo, por ejemplo, supongamos un proyecto de duración 10 semanas ($t = 10$). El esfuerzo *(story point)* asociado a una tarea de desarrollo al principio del proyecto ($t_0 = 1$) no será el mismo que el de realizar una tarea similar más adelante ($t_1 = 5$), dado que el equipo de desarrollo habrá ganado experiencia tanto con el proyecto como con las herramientas y lenguajes de desarrollo, por lo que el esfuerzo de estas dos tareas va a variar dependiendo del momento en el que se realicen. Esto implica que la equivalencia: 1 *story point* = 1 hora sea errónea. Para gestionar los *story points* de las tareas del proyecto, se va a utilizar un *power-up* de Trello llamado **Agile Tools**[1].

Esta herramienta nos permite configurar y gestionar el rango de valores de puntos de historia que se van a asignar a una determinada tarea. El rango de valores que utilizaremos vendrá dado por la sucesión de Fibonnaci hasta el número 13 (1,2,3,5,8,13).

Además de esto, la herramienta realiza una asignación de colores en función del valor de la

[1] https://getcorrello.com/AgileTools.

complejidad que le asignemos a la tarea: 1) verde para tareas sencillas (complejidad 1-3); 2) amarillo para tareas de complejidad media (5 y 8) y 3) naranja para las tareas de mayor dificultad (complejidad 13). De esta forma podemos observar de una forma rápida y descriptiva el esfuerzo que va a suponer realizar una determinada tarea.

Otra funcionalidad que ofrece esta herramienta relacionada con los story points asociados a las tareas, es la posibilidad de resumir el número total de story points que hay en cada una de las columnas del tablero. Esta funcionalidad es útil cuando el número de tareas es elevado ya que permite ver el esfuerzo ligado a cada columna pudiendo así ver el avance del proyecto en un momento determinado.

2.3. Herramientas Utilizadas

Para llevar a cabo la elaboración del proyecto se han utilizado las siguientes herramientas:

1) **GitLab:** Servicio web gratuito de código abierto centralizado en el manejo de control de versiones utilizando Git como base. GitLab fue lanzado en Octubre de 2011 y permite el alojamiento de repositorios en los que se podrá llevar a cabo el manejo de las versiones y de los errores. Además de esto, GitLab permite también el alojamiento de wikis y la creación de grupos tanto a nivel de equipo como a nivel de repositorios.

2) **Jupyter Notebook:** Jupyter es un proyecto de código abierto creado en 2014 a partir de IPython. Entre las herramientas de computación interactiva se destaca Jupyter Notebook la cual se encuentra constituida por una aplicación cliente-servidor que permite crear y compartir documentos web en formato JSON siguiendo un esquema de lista ordenada de celda con entradas y salidas. Estas celdas soportan tanto código en Python como código en R además de texto en formato Markdown. El nombre de este proyecto viene en honor de los tres principales lenguajes que soporta: Julia, Python y R.

3) **Overleaf:** Herramienta colaborativa que permite la creación y edición de documentos en línea utilizando el lenguaje de marcado LaTeX.

4) **Visual Studio Code:** Editor de código fuente desarrollado por Microsoft y lanzado de forma oficial en el año 2016. Visual Studio Code proporciona herramientas de depuración de código y soporta la mayoría de los lenguajes de programación utilizados gracias a su gran catálogo de extensiones. Además de esto cuenta con una gran integración con Git facilitando la labor de llevar a cabo un buen control de versiones.

5) **WinSCP:** Herramienta gratuita de código abierto que proporciona un cliente SFTP gráfico para Windows utilizando el protocolo de acceso remoto SSH. Esta herramienta ha sido utilizada para realizar el manejo y transmisión de archivos entre la máquina virtual y la máquina local.

2.4. Tecnologías Utilizadas

Las tecnologías que se han utilizado para llevar a cabo la realización del proyecto han sido las siguientes:

1) **CSS:** Lenguaje de diseño gráfico que permite crear y definir la apariencia de los documentos basados en lenguajes de marcado (HTML, XHTML, XML, etc). Su uso principal se realiza para mejorar la apariencia de las interfaces web del lado cliente. Es considerado uno de los 3 pilares en el desarrollo de páginas web en el lado del cliente junto con los lenguajes HTML y JavaScript.

2) **Dash:** Framework de Python de código libre para la generación de aplicaciones web y *dashboards* interactivos utilizando para ello Flask, Plotly y ReactJS. Las aplicaciones Dash se ejecutan y renderizan en el navegador por lo que es posible desplegar la aplicación en un servidor y compartir su contenido con varias personas, pudiendo estas realizar distintas interacciones sobre estos dashborads de forma independiente ya que cada *dashboard* es independiente en cada navegador. Además de esto Dash es un framework multiplataforma y es compatible con Boostrap.

3) **Git:** Sistema de control de versiones que gestiona y registra todos los cambios que se realizan en los archivos situados en el repositorio local en el ordenador, mejorando el control que se tiene sobre estos y permitiendo navegar entre las distintas versiones que hayan tenido esos archivos.

4) **Matplotlib y Seaborn:** Matplotlib y Seaborn son bibliotecas especializadas en la creación de gráficos a alto nivel utilizando Python como lenguaje principal. Estas bibliotecas disponen de integración directa tanto con Pandas como con Numpy, permitiendo realizar visualizaciones de una forma rápida y elegante de datos representados en las principales estructuras de datos de estas bibliotecas (DataFrames, Arrays y Matrices) lo cual la hace indispensable para labores de visualización.

5) **NumPy:** NumPy es una biblioteca de código abierto para Python que incluye herramientas matemáticas más avanzadas para realizar operaciones sobre vectores o matrices.

6) **Pandas:** Pandas es una biblioteca de código abierto para Python creada como extensión de Numpy que permite la manipulación y el análisis de datos, utilizando estructuras propias (DataFrames). Es una de las bibliotecas más utilizadas ya que permite realizar de forma rápida y eficiente inserciones, eliminaciones, agrupaciones de datos así como otras características básicas a la hora de trabajar con estos: obtención de valores estadísticos (media, mediana, cuartiles, percentiles, etc), operaciones de ordenación y filtrado de datos, manejo de series temporales, etc.

7) **Python:** Lenguaje de programación multiparadigma diseñado por Guido van Rossum y lanzado en 1991, cuya filosofía hace hincapié en la legibilidad de su código. Python se encuentra administrado por la Python Software Foundation bajo una licencia de código abierto. Es considerado como uno de los 5 lenguajes de programación más importantes en la actualidad, contando con una gran comunidad que no deja de desarrollar bibiliotecas y módulos que enriquezcan su funcionalidad. Python es uno de los lenguajes más polifacéticos ya que con el es posible desarrollar aplicaciones web, aplicaciones de escritorio, juegos, y es el lenguaje más utilizado para tareas de análisis de datos y el aprendizaje automático, gracias a la gran integración con bibliotecas estadísticas (Numpy, SciPy), de visualización (Matplotlib, Seaborn), de manejo de datos (Pandas) o de aprendizaje automático (Scikit-learn, Tensorflow, Keras, Pytorch, etc).

8) **Scikit-learn:** Biblioteca de código abierto que proporciona herramientas para el uso de aprendizaje automático en el lenguaje de programación Python. Esta biblioteca incluye varios algoritmos de aprendizaje automático tanto supervisado como no supervisado para labores tanto de clasificación como de regresión. Además de esto, incluye las principales funciones que se utilizan a la hora de crear modelos de aprendizaje automático como la división estratificada de datos, métricas para evaluar los modelos o funciones para el entrenamiento de los mismos. Asimismo, dispone de integración directa con Pandas y NumPy, lo que permite una mejor robustez a la hora de realizar operaciones sobre los datos y los modelos.

Capítulo 3

Estimación y Presupuestos

3.1. Estimación

Debido a que este proyecto se realiza como parte de la asignatura de Trabajo de Fin de Grado, tanto el alcance como la duración del proyecto han de ajustarse al número de horas estipulado por la asignatura (12 créditos ECTS) los cuales se traducen en 300h de implicación.

Para llevar a cabo el proyecto, se ha realizado una división de este en 7 *sprints*, con una duración de 3 semanas cada uno, siendo el inicio tanto del proyecto como del primer *sprint* el día 5 de febrero de 2020 y concluyendo el último *sprint* el día 1 de julio de 2020. Las historias de usuario obtenidas para cumplir con los objetivos marcados para el proyecto son:

- **H-01:** Estudio del dominio del proyecto.

  Esta historia se basa en la obtención tanto de los conocimientos necesarios para poder abordar el tema principal a tratar a lo largo del proyecto (pistas, configuración de las pistas, gestión del tráfico aéreo, etc), como en estudiar y analizar otras propuestas similares que se han realizado con anterioridad, con el objetivo de consolidar una ruta de trabajo sobre la desarrollar satisfactoriamente el proyecto (ver Tabla 3.1).

- **H-02:** Carga, preprocesamiento y normalización de los datos.

  En esta historia se comprenden todas aquellas tareas que tienen como eje central los datos: carga del dataset, operaciones de limpieza y tratamiento de datos ausentes, aplicación de operaciones de normalización y codificación, eliminación de características innecesarias y, finalmente, la exportación del conjunto de datos final. El objetivo de esta historia consiste en refinar el conjunto de datos inicial para lograr una mayor eficacia con el modelo de aprendizaje automático (ver Tabla 3.2).

- **H-03:** Selección y mejora de los modelos.

  Una vez llevado a cabo el proceso de normalización y adaptación de los datos, es el momento de analizar y observar el comportamiento de los modelos de aprendizaje automático sobre estos datos, con la finalidad de identificar aquellos modelos que obtengan mejores resultados y finalmente realizar un proceso de optimización y refinamiento de los mismos con el objetivo de mejorar los resultados obtenidos con la configuración base de este (ver Tabla 3.3).

- **H-04:** Evaluación y selección del modelo final.

Seguidamente, tras haber implementado y mejorado varios modelos de aprendizaje automático, el siguiente paso es evaluar cada uno de los modelos obtenidos, aplicando distintas técnicas de evaluación para comprobar que este se comporta correctamente, que no sufre de sobreajuste y obtener sus métricas principales (*precision*, *recall* y *f1*). Finalmente, y en base a las conclusiones obtenidas, se selecciona el modelo final que se utilizará para llevar a cabo la tarea de clasificación de la configuración de las pistas (ver Tabla 3.4).

- **H-05:** Desarrollo e integración del modelo final en el *Dashboard*.

  Esta historia tiene como objetivo la generación de un *dashboard* interactivo que permita al usuario tanto interactuar con el modelo de aprendizaje automático (introduciendo la información asociada a un vuelo para que el modelo lleve a cabo la predicción), como visualizar distintas métricas asociadas a estos modelos: *precision*, curva de validación, matriz de confusión y validación cruzada (ver Tabla 3.5).

- **H-06:** Desarrollo de la memoria del proyecto.

  Cada una de las fases de las que consta el proyecto deberá ser plasmada en la memoria de una forma clara, concisa y siguiendo la estructura de la misma proporcionada por la titulación. Dicha memoria tendrá plasmada la contextualización del tema a abordar, la planificación, estimación, presupuestos y organización del proyecto, y finalmente, todo el proceso desarrollado durante las fases de implementación de la propuesta, incluyendo las conclusiones finales que se han obtenido de la misma (ver Tabla 3.6).

Para cada una de las historias descritas, se ha realizado un desglose en tareas más pequeñas cuya finalización se materializa en la consecución de los objetivos y criterios de la historia. La dificultad de cada una de ellas se ha obtenido mediante una estimación utilizando la técnica de *estimación por póquer* [25], empleando como escala la sucesión de Fibonacci. Gracias a esto se consigue establecer a cada una un determinado valor de complejidad de una forma más sencilla y permitiendo un reparto más equivalente de estas entre los sucesivos *sprints* que conforman el proyecto.

A continuación se muestra el desglose en tareas de las historias anteriormente descritas:

Tarea	Descripción	Dificultad
T1.1	**Estudio de la gestión del tráfico aéreo en la actualidad**	*1 Punto de Historia*
	Esta tarea consiste en realizar un estudio inicial de la situación actual del tráfico aéreo que sirva como partida para llevar a cabo el proceso de entendimiento del dominio.	
T1.2	**Estudio de la utilización de pistas y su configuración**	*1 Punto de Historia*
	El objetivo de esta tarea consiste en llevar a cabo un estudio de como se lleva a cabo la utilización de las pistas en un aeropuerto, así como los criterios que rigen tanto la utilización de una configuración u otra como el cambio de las mismas en un momento determinado.	
T1.3	**Estudio de los factores que afectan a la elección de la configuración**	*1 Punto de Historia*
	Esta tarea tiene como objetivo el realizar un estudio sobre todos aquellos factores que afectan y determinan la elección de una determinada configuración frente a otra, así como los criterios que se llevan a cabo para realizar esta elección.	
T1.4	**Estudio de los sistemas de control y motorización de un vuelo**	*1 Punto de Historia*
	Esta tarea tiene como objetivo aprender acerca de los sistemas que se utilizan para controlar y monitorizar cada uno de los vuelos, ver que datos se recogen, la realización que existen entre estos, como se realiza la transmisión de datos para llevar a cabo la monitorización, etc.	
T1.5	**Estudio de propuestas similares**	*3 Puntos de Historia*
	El objetivo de esta tarea consiste en realizar un estudio completo de propuestas similares realizadas con anterioridad (correspondiente al estudio del estado del arte), obteniendo una visión de que cosas se han estado probado en esta rama y los resultado obtenidos.	

Tabla 3.1: Desglose de las tareas asociadas a la Historia 1 del Proyecto.

Tarea	Descripción	Dificultad
T2.1	**Carga e interpretación del conjunto de datos**	*1 Punto de Historia*
	Esta tarea tiene como objetivo llevar a cabo el proceso de carga e interpretación de cada una de las características del conjunto de datos inicial, solucionando posibles incidencias que puedan ocurrir durante el mismo y comprobando que la carga de todos los datos se realiza satisfactoriamente.	
T2.2	**Limpieza de los datos**	*2 Puntos de Historia*
	Esta tarea tiene como objetivo la realización de la limpieza de los datos, realizando diferentes transformaciones sobre los mismos que permitan ser utilizados en etapas posteriores del proyecto así como solventar peculiaridades que pudieran existir (datos nulos, formatos incorrectos, etc).	
T2.3	**Normalización de los datos**	*2 Puntos de Historia*
	El objetivo de esta tarea consiste en realizar una normalización sobre los datos, de tal forma que las variables cualitativas que existan en estos se sustituyan por variables cuantitativas sin perder la información que estas representan, consiguiendo así que los resultados finales mejoren con respecto a la omisión de este proceso.	
T2.4	**Exportación del conjunto de datos final**	*1 Punto de Historia*
	Esta tarea tiene como objetivo guardar una versión refinada de los datos originales de acuerdo a las operaciones de transformación descritas para evitar tener que procesarlos de nuevo en cada uso.	

Tabla 3.2: Desglose de las tareas asociadas a la Historia 2 del Proyecto.

Tarea	Descripción	Dificultad
T3.1	**Prueba inicial de los modelos**	*3 Puntos de Historia*
	Esta tarea tiene como objetivo el realizar una prueba de los principales algoritmos de aprendizaje automático destinados a tareas de clasificación, con el objetivo de observar el comportamiento base de cada uno y comprobar si son adecuados para esta tarea o si se han de descartar.	
T3.2	**Refinamiento de los modelos**	*5 Puntos de Historia*
	El objetivo de esta tarea consiste en realizar modificaciones sobre los distintos hiperparámetros que conforman los modelos utilizados en la tarea T3.1, con la finalidad de mejorar el comportamiento de los mismos.	

Tabla 3.3: Desglose de las tareas asociadas a la Historia 3 del Proyecto.

Tarea	Descripción	Dificultad
T4.1	**Analisis de los resultados obtenidos**	*3 Puntos de Historia*
	El objetivo de esta tarea consiste en realizar un análisis de los resultados obtenidos para cada uno de los modelos tras haber realizado el refinamiento de los mismos, obteniendo las métricas correspondientes de los mismos y analizando su comportamiento definitivo.	
T4.2	**Elección y exportación del modelo final**	*2 Puntos de Historia*
	Esta tarea tiene como objetivo determinar (a partir de los modelos finales y del análisis de los mismos realizado en la tarea T4.1), aquel modelo cuyos resultados sean mejores, justificando los motivos por los cuales se ha llegado a esa conclusión y finalmente exportando el modelo seleccionado.	

Tabla 3.4: Desglose de las tareas asociadas a la Historia 4 del Proyecto.

Tarea	Descripción	Dificultad
T5.1	**Creación del dashboard**	*5 Puntos de Historia*
	Esta tarea tiene como objetivo el desarrollo del *dashboard* que se utilizará para mostrar de una forma visual e interactiva tanto los resultados obtenidos a lo largo de las fases de análisis de los modelos, como permitir utilizar el modelo final para realizar predicciones con los datos que el usuario inserte en dicho *dashboard*.	
T5.2	**Integración del modelo en el dashboard**	*2 Puntos de Historia*
	El objetivo de esta tarea consiste en integrar el modelo en el *dashboard* para que este sea capaz de realizar predicciones gracias a los datos que inserta el usuario a través del formulario del *dashboard*.	

Tabla 3.5: Desglose de las tareas asociadas a la Historia 5 del Proyecto.

Tarea	Descripción	Dificultad
T6.1	**Documentación de la introducción y el contexto del proyecto**	*2 Puntos de Historia*
	Esta tarea consiste en redactar la parte introductoria del proyecto, así como la contextualización del entorno sobre el que se está trabajando.	
T6.2	**Documentación de la planificación, estimación y presupuestación del proyecto**	*1 Punto de Historia*
	El objetivo de esta tarea consiste en documentar toda la parte referente a la organización y gestión llevada a cabo a lo largo del proyecto.	
T6.3	**Documentación de la operaciones realizadas para la carga, limpieza y normalización de los datos**	*1 Punto de Historia*
	Esta tarea consiste en realizar la documentación relativa a toda la parte de los datos, desde su estado inicial, hasta el estado final de los mismos, pasando por todas las operaciones de transformación y normalización realizadas sobre estos.	
T6.4	**Documentación del procedimiento y resultados obtenidos en el análisis inicial de los modelos de aprendizaje automático**	*1 Punto de Historia*
	El objetivo de esta tarea consiste en documentar todo lo relacionado a la prueba inicial de modelos de aprendizaje automático (qué modelos se han utilizado, los datos empleados, resultados iniciales, etc), además de un análisis de los resultados que se han obtenido en los mismos.	
T6.5	**Documentación del proceso de refinamiento de los modelos de aprendizaje automático**	*1 Punto de Historia*
	Esta tarea consiste en exponer qué modificaciones se han llevado a cabo a los distintos modelos de aprendizaje automático, y cómo han afectado estos al rendimiento, realizando una comparativa entre el modelo con estos cambios y el modelo sin ellos.	
T6.6	**Documentación de los resultados obtenidos en el modelo de aprendizaje automático final**	*1 Punto de Historia*
	El objetivo de esta tarea consiste en documentar tanto los resultados obtenidos en el modelo final como un análisis de los mismos de tal forma que se reflejen todas aquellas cuestiones importantes en dichos resultados, así como la interpretación de estos.	
T6.7	**Documentación de la utilización del dashboard**	*1 Punto de Historia*
	El objetivo de esta tarea consiste en realizar un pequeño manual de usuario en el que se facilite la información necesaria para utilizar todas las características que proporciona el *dashboard desarrollado*.	
T6.8	**Documentación de las conclusiones y trabajo futuro**	*1 Punto de Historia*
	Esta tarea tiene como objetivo el realizar una síntesis de todo lo aprendido así como un balance de los resultados obtenidos sobre los objetivos propuestos para el proyecto.	
T6.9	**Documentación del glosario, anexos y documentación adjunta al proyecto**	*1 Punto de Historia*
	Esta tarea tiene como objetivo realizar la documentación de aquellas partes complementarias a la memoria final, tales como el anexo, el glosario o cualquier documentación que pueda ser importante para aclarar alguna parte concreta del proyecto.	

Tabla 3.6: Desglose de las tareas asociadas a la Historia 6 del Proyecto.

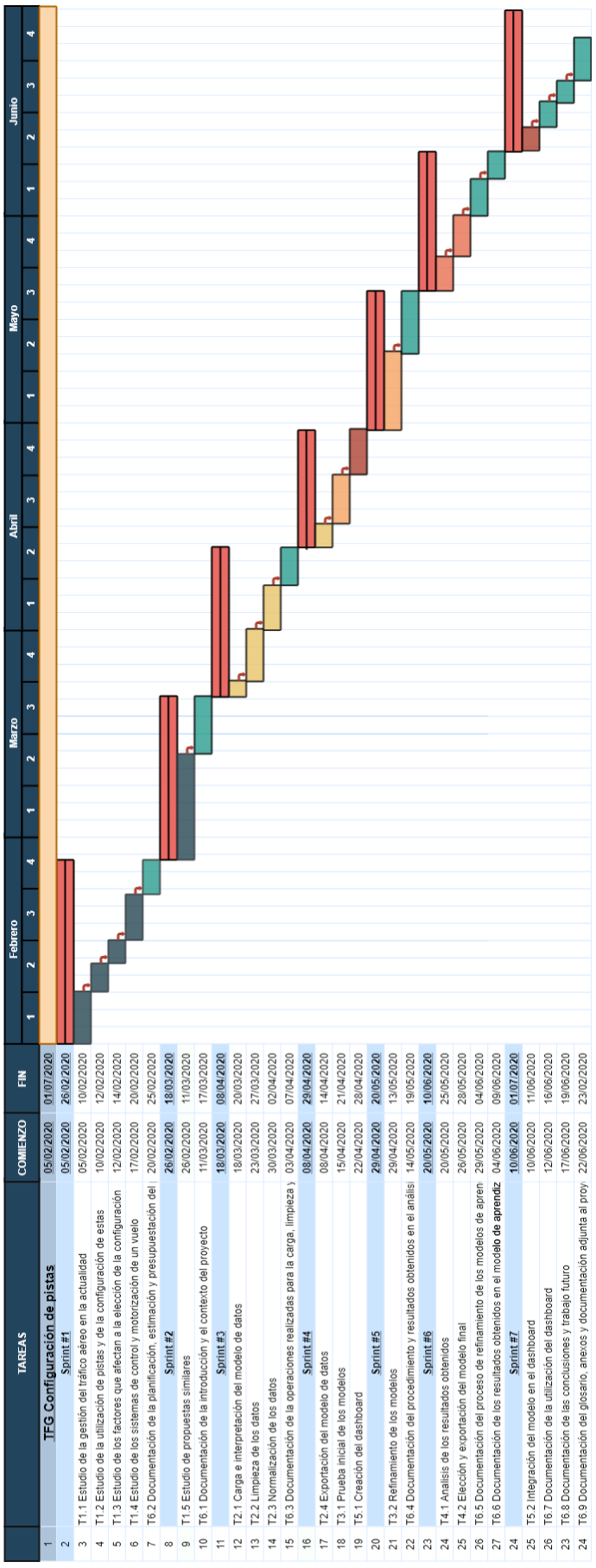
Para llevar a cabo la consecución del proyecto, se han estimado un total de 61 puntos de historia a distribuir entre cada uno de los 7 *sprints* en los que se encuentra dividido el proyecto. A cada uno de estos *sprints* se le ha asignado un total de 43 horas de trabajo efectivo, de tal forma que el total de todos ellos equivale al número de horas estipulado para el Trabajo Fin de Grado (43 horas · 7 *sprints* = 301 horas). A pesar de que no existe una equivalencia directa, objetiva y estandarizada entre el número de horas y los puntos de historia, es posible hacer una distribución de las tareas entre los distintos *sprints*. Para ello, se le ha asignado al primer *sprint* (basándonos en nuestra experiencia previa) un conjunto de tareas por valor de X Puntos de Historia. De esta forma, durante la primera *RetroPlanning*, se analizan cuántos Puntos de Historia del total asignado a dicho *sprint* han sido completados, consiguiendo así obtener (de forma aproximada) el número de Puntos de Historia que el equipo de desarrollo es capaz de realizar en un *sprint*. De este modo se obtiene de forma aproximada la relación que existe entre 1 Punto de Historia y 1 hora para ese equipo de desarrollo en el proyecto, adaptando las tareas del resto de *sprints* a la capacidad de desarrollo por parte del equipo.

Por otra parte, en el caso de que al terminar el *sprint* no se hubiera conseguido finalizar todas las tareas asignadas o se hubieran completado muy rápido, significa que la estimación no ha sido correcta, por lo que, en posteriores *sprint* será necesario asignar o eliminar tareas para ajustarse su duración. Como se puede ver en la Tabla 3.7, la diferencia entre el número de Puntos de Historia asignados a cada *sprint* es muy pequeña (excepto en el último *sprint* debido a que ya no había más tareas para asignar). Esto se traduce en una buena estimación, planificación y distribución de la carga de trabajo a lo largo del desarrollo del proyecto.

A continuación se muestra tanto la distribución de tareas en los *sprints* que conforman el proyecto (ver Tabla 3.7) como el diagrama de Gantt correspondiente a la planificación temporal descrita (ver Figura 3.1). Para facilitar la visualización y no ser redundante con tareas periódicas, en las tablas y figuras no se detallan las tareas correspondientes a las distintas ceremonias y reuniones que se realizan en un proyecto *Scrum* de UVAGILE: *RetroPlanning*, *Weeklies* o reuniones con los tutores de proyecto.:

Sprint	**Inicio**	**Fin**	**Tareas**	**Total puntos de historia**
Sprint #1	05/02/2020	26/02/2020	T1.1; T1.2; T1.3; T1.4; T6.2	9
Sprint #2	26/02/2020	18/03/2020	T1.5; T6.1	8
Sprint #3	18/03/2020	08/04/2020	T2.1; T2.2; T2.3 T6.3	9
Sprint #4	08/04/2020	29/04/2020	T2.4; T3.1; T5.1	9
Sprint #5	29/04/2020	20/05/2020	T3.2; T6.4	10
Sprint #6	20/05/2020	10/06/2020	T4.1; T4.2; T6.5; T6.6	9
Sprint #7	10/06/2020	01/07/2020	T5.2; T6.7; T6.8; T6.9	7

Tabla 3.7: Distribución de tareas por *sprints*.



	TAREAS	COMIENZO	FIN
1	**TFG Configuración de pistas**	05/02/2020	01/07/2020
2	**Sprint #1**	05/02/2020	26/02/2020
3	T1.1 Estudio de la gestión del tráfico aéreo en la actualidad	05/02/2020	10/02/2020
4	T1.2 Estudio de la utilización de pistas y de la configuración de estas	10/02/2020	12/02/2020
5	T1.3 Estudio de los factores que afectan a la elección de la configuración	12/02/2020	14/02/2020
6	T1.4 Estudio de los sistemas de control y motorización de un vuelo	17/02/2020	20/02/2020
7	T6.2 Documentación de la planificación, estimación y presupuestación del	20/02/2020	25/02/2020
8	**Sprint #2**	26/02/2020	18/03/2020
9	T1.5 Estudio de propuestas similares	26/02/2020	11/03/2020
10	T6.1 Documentación de la introducción y el contexto del proyecto	11/03/2020	17/03/2020
11	**Sprint #3**	18/03/2020	08/04/2020
12	T2.1 Carga e interpretación del modelo de datos	18/03/2020	20/03/2020
13	T2.2 Limpieza de los datos	23/03/2020	27/03/2020
14	T2.3 Normalización de los datos	30/03/2020	02/04/2020
15	T6.3 Documentación de la operaciones realizadas para la carga, limpieza y	03/04/2020	07/04/2020
16	**Sprint #4**	08/04/2020	29/04/2020
17	T2.4 Exportación del modelo de datos	08/04/2020	14/04/2020
18	T3.1 Prueba inicial de los modelos	15/04/2020	21/04/2020
19	T5.1 Creación del dashboard	22/04/2020	28/04/2020
20	**Sprint #5**	29/04/2020	20/05/2020
21	T3.2 Refinamiento de los modelos	29/04/2020	13/05/2020
22	T6.4 Documentación del procedimiento y resultados obtenidos en el análisi	14/05/2020	19/05/2020
23	**Sprint #6**	20/05/2020	10/06/2020
24	T4.1 Análisis de los resultados obtenidos	20/05/2020	25/05/2020
25	T4.2 Elección y exportación del modelo final	26/05/2020	28/05/2020
26	T6.5 Documentación del proceso de refinamiento de los modelos de apren	29/05/2020	04/06/2020
27	T6.6 Documentación de los resultados obtenidos en el modelo de aprendiz	04/06/2020	09/06/2020
24	**Sprint #7**	10/06/2020	01/07/2020
25	T5.2 Integración del modelo en el dashboard	10/06/2020	11/06/2020
26	T6.7 Documentación de la utilización del dashboard	12/06/2020	16/06/2020
23	T6.8 Documentación de las conclusiones y trabajo futuro	17/06/2020	19/06/2020
24	T6.9 Documentación del glosario, anexos y documentación adjunta al proy	22/06/2020	23/02/2020

Figura 3.1: Diagrama de Gantt del proyecto.

3.2. Presupuestos

Una vez realizada la estimación de la planificación temporal que se llevará a cabo a lo largo del proyecto, podemos realizar una estimación del presupuesto, incluyendo todas las herramientas (hardware y software) que se han empleado para su desarrollo, independientemente del coste asociado a las mismas.

Por la parte hardware, toda la implementación del código se llevará a cabo en un ordenador personal, que por especificaciones puede considerarse de gama media, (Procesador I5-8250U, 64 bits, 8GB de RAM y 500 GB SSD), utilizando herramientas o bien de software libre o bien herramientas de pago con suscripciones gratuitas debido al rol de estudiante universitario. No obstante, toda la parte pesada de prueba, refinamiento y generación de modelos, se ejecutará sobre una máquina virtual proporcionada por el Departamento de Informática de la Universidad de Valladolid, cuya utilización no supone un coste directo para el proyecto pero se ha tenido en cuenta (dado que no se conoce el coste por hora del uso de esta máquina virtual, el cálculo asociado a la misma se tomará basándonos en una máquina virtual de *Amazon Web Services* con unas especificaciones similares). Las especificaciones de esta máquina son: Intel de 8 núcleos 64 bits, 16 GB de RAM y 30GB de almacenamiento HDD. Del mismo modo, debido tanto al uso de esta máquina virtual, utilización de herramientas de salvaguarda del proyecto, consulta de documentación, etc, es crucial disponer de una conexión a Internet que permita realizar todas estas acciones, por lo que este gasto se incluirá en el proyecto. Finalmente, debido a la utilización de material electrónico, es necesario incluir también en el presupuesto el coste asociado al gasto energético generado por la máquina personal, tomando para ello el número de horas aproximadas que se ha utilizado junto con el consumo de la misma.

La Tabla 3.8 muestra un desglose de cada una de las herramientas utilizadas, así como el gasto asociado a cada una de ellas:

Herramienta	**Coste (mes)**	**Vida útil**	**% de uso**	**Total**
Ordenador Personal	800€ (coste total)	5 años	8,33 %	66,64€
Internet	20€	5 meses	100 %	100€
Coste Energético	7,2€	5 meses	100 %	37,5€
Máquina Virtual	64,64€	3 meses	60 %	193,92€
GitLab	0€	–	–	0€
Visual Studio Code	0€	–	–	0€
Overleaf	0€	–	–	0€
Anaconda	0€	–	–	0€
Trello	0€	–	–	0€
Slack	0€	–	–	0€
TOTAL				**398,06€**

Tabla 3.8: Costes asociados a herramientas y recursos hardware.

En cuanto a los costes asociados a los recursos humanos, a pesar de que el trabajo ha sido realizado por una única persona, dicha persona ha tomado distintos roles durante el desarrollo del mismo, por lo que el coste asociado a la parte de recursos humanos, ha de tener en cuenta este hecho. Para llevar a cabo estos cálculos, se ha supuesto la dedicación habitual en una empresa en

la actualidad (40 horas semanales, o lo que es lo mismo, 160 horas el mes). La Tabla 3.9 muestra los distintos roles que se han tomado así como las horas de dedicación y el coste asociado a cada una:

Rol	Salario (mes)	Horas	Total
Jefe de Proyecto	3.100€	40	775€
Analista de Requisitos	2.300€	67	963,12€
Desarrollador Python	2.100€	142	1.863,75€
Desarrollador Web	1.800€	56	630€
TOTAL			**4.231,87€**

Tabla 3.9: Costes asociados a recursos humanos. Fuente: *Linkedin Salary (visitado 20-04-2020).*

Por tanto, el coste total previsto del proyecto es de **4.629,93€** de los cuales 398.06€ vienen dados por los recursos hardware y 4.231,87€ por los recursos humanos.

3.3. Balance final del proyecto

3.3.1. Planificación temporal

Durante el primer *sprint* del proyecto se cumplió de forma satisfactoria la planificación temporal establecida, finalizando las tareas relativas a la obtención de los conocimientos básicos necesarios para poder llevar a cabo el desarrollo del proyecto (T1.1, T1.2, T1.3 y T1.4) y realizando la planificación temporal y económica que se seguirá a lo largo del desarrollo del proyecto, tal y como se establecía en la tarea T6.2.

A lo largo del segundo *sprint*, se llevó a cabo la última tarea referente a la historia H-01, dando por cerrada y consolidada la parte correspondiente a la obtención de los conocimientos necesarios para abordar de forma satisfactoria el proyecto. Finalmente, durante la última semana del *sprint*, se plasmó en la documentación los conocimientos adquiridos a lo largo de las tareas de la historia H-01, avanzando así con la historia H-06 enfocada en el desarrollo de la memoria del proyecto.

El tercer *sprint* transcurrió según lo esperado de acuerdo a la planificación establecida, completando en tiempo y forma las tareas relativas a la carga, limpieza y normalización de los datos (T2.1, T2.2, T2.3). Del mismo modo, se finalizó la tarea T6.3, consistente en realizar la documentación referente al procedimiento llevado a cabo para generar el conjunto de datos final a partir del original.

A lo largo de la *RetroPlanning* correspondiente al cuarto *sprint*, se produjo un gran imprevisto debido a que no se disponía de información explícita sobre la configuración de las pistas en el momento de la realización la maniobra, por lo que no se pudo iniciar la tarea relativa a la prueba de los modelos de aprendizaje automático (T3.1) debido a que no se disponía de la característica a predecir. Esto supuso la ampliación de la historia H-02 así como la reestructuración de las tareas que la conforman tal y como se muestra en la Tabla 3.11. Esta nueva tarea supuso un reajuste en la planificación inicial, tanto en el cuarto *sprint* como en los posteriores. Finalmente el *sprint* concluyó con la consecución de las tareas T2.5 (nueva), T2.4 y T5.1, trasladando la tarea T3.1 al siguiente *sprint*.

El comienzo del quinto *sprint* se vió restringido debido a la introducción de una nueva tarea al

proyecto, suponiendo un retraso en la planificación realizada y obligando a realizar un reajuste en los sucesivos *sprints*. Debido a estos hechos, se pospuso la realización de la tarea T6.4 a *sprints* posteriores y se llevaron a cabo de forma satisfactoria las tareas T3.1 (correspondiente al *anterior*) y T3.2, dando por finalizada la historia H-03, relativa a la creación y optimización de los distintos modelos de aprendizaje automático, cumpliendo con los objetivos asociados a la misma.

Al igual que ocurrió con el quinto *sprint*, el alcance de este *sprint* se vió condicionado por el retraso producido por la postergación de las tareas en *sprints* anteriores. Es por ello por lo que en este *sprint* se ha llevado a cabo la tarea T6.4 (planificada para el quinto *sprint*) y se aplazó la tarea T6.6 (planificada para este *sprint*). En cuanto a la finalización de los objetivos fijados, se concluyeron satisfactoriamente las tareas T4.1 y T4.2, consolidando la parte correspondiente a la evaluación y selección del modelo final de aprendizaje automático, concluyendo así con una de las características principales a tratar en el proyecto. Finalmente también se llevaron a cabo las tareas T6.4 y T6.5 correspondientes a la documentación del proceso de refinamiento y análisis de los resultados obtenidos a lo largo de la optimización y selección de los modelos de aprendizaje automático utilizados.

Finalmente, el séptimo y último *sprint* se llevó a cabo según la planificación inicial salvo por la inclusión de la tarea T6.6, la cual pertenecía al sexto *sprint* y no pudo abordarse debido a los inconvenientes ya descritos. No obstante, tanto esta tarea como las planificadas para este *sprint* fueron terminadas en tiempo y forma, finalizando así las historias H-05 (correspondiente con el desarrollo e implementación de la herramienta software) y H-06 (correspondiente a la realización de la documentación del proyecto).

En este punto cabe destacar el impacto en cuanto a la planificación que ha supuesto la introducción de la Tarea T2.5 (la cuál no se encontraba dentro de la planificación inicial del proyecto). En términos generales, no ha supuesto ninguna variación temporal, ya que, tal y como se encontraba estructurada la planificación inicial, el último *sprint* disponía de una menor carga de trabajo debido a que se habían completado todas las tareas del proyecto. Esta particularidad ha permitido redistribuir las tareas de los *sprints* #5, #6 y #7 (ver Tabla 3.10), haciendo innecesaria la realización de horas extras o la extensión del tiempo del proyecto, consiguiendo finalizar en las fechas estipuladas en la planificación inicial.

Sprint	**Inicio**	**Fin**	**Tareas**	**Total puntos de historia**
Sprint #1	05/02/2020	26/02/2020	T1.1; T1.2; T1.3; T1.4; T6.2	9
Sprint #2	26/02/2020	18/03/2020	T1.5; T6.1	8
Sprint #3	18/03/2020	08/04/2020	T2.1; T2.2; T2.3 T6.3	9
Sprint #4	08/04/2020	29/04/2020	T2.4; T2.5; T5.1	9
Sprint #5	29/04/2020	20/05/2020	T3.1; T3.2	11
Sprint #6	20/05/2020	10/06/2020	T4.1; T4.2; T6.4; T6.5	9
Sprint #7	10/06/2020	01/07/2020	T5.2; T6.6; T6.7; T6.8; T6.9	9

Tabla 3.10: Distribución actualizada de tareas por *sprints*.

En la Figura 3.2 se refleja el diagrama de Gantt asociado a la nueva planificación adoptada tras las modificaciones llevadas a cabo.

Tarea	Descripción	Dificultad
T2.1	**Carga e interpretación del conjunto de datos**	*1 Punto de Historia*
	Esta tarea tiene como objetivo llevar a cabo el proceso de carga e interpretación de cada una de las características del conjunto de datos inicial, solucionando posibles incidencias que puedan ocurrir durante el mismo y comprobando que la carga de todos los datos se realiza satisfactoriamente.	
T2.2	**Limpieza de los datos**	*2 Puntos de Historia*
	Esta tarea tiene como objetivo la realización de la limpieza de los datos, realizando diferentes transformaciones sobre los mismos que permitan ser utilizados en etapas posteriores del proyecto así como solventar peculiaridades que pudieran existir (datos nulos, formatos incorrectos, etc).	
T2.3	**Normalización de los datos**	*2 Puntos de Historia*
	El objetivo de esta tarea consiste en realizar una normalización sobre los datos, de tal forma que las variables cualitativas que existan en estos se sustituyan por variables cuantitativas sin perder la información que estas representan, consiguiendo así que los resultados finales mejoren con respecto a la omisión de este proceso.	
T2.4	**Exportación del conjunto de datos final**	*1 Punto de Historia*
	Esta tarea tiene como objetivo guardar una versión refinada de los datos originales de acuerdo a las operaciones de transformación descritas para evitar tener que procesarlos de nuevo en cada uso.	
T2.5	**Obtención de las configuraciones a partir de la utilización de las pistas**	*3 Puntos de Historia*
	El objetivo de esta tarea consiste en la obtención de la configuración de las pistas para cada intervalo temporal de 30 minutos, a partir del conjunto de datos original.	

Tabla 3.11: Desglose actualizado de las tareas asociadas a la Historia 2 del Proyecto.

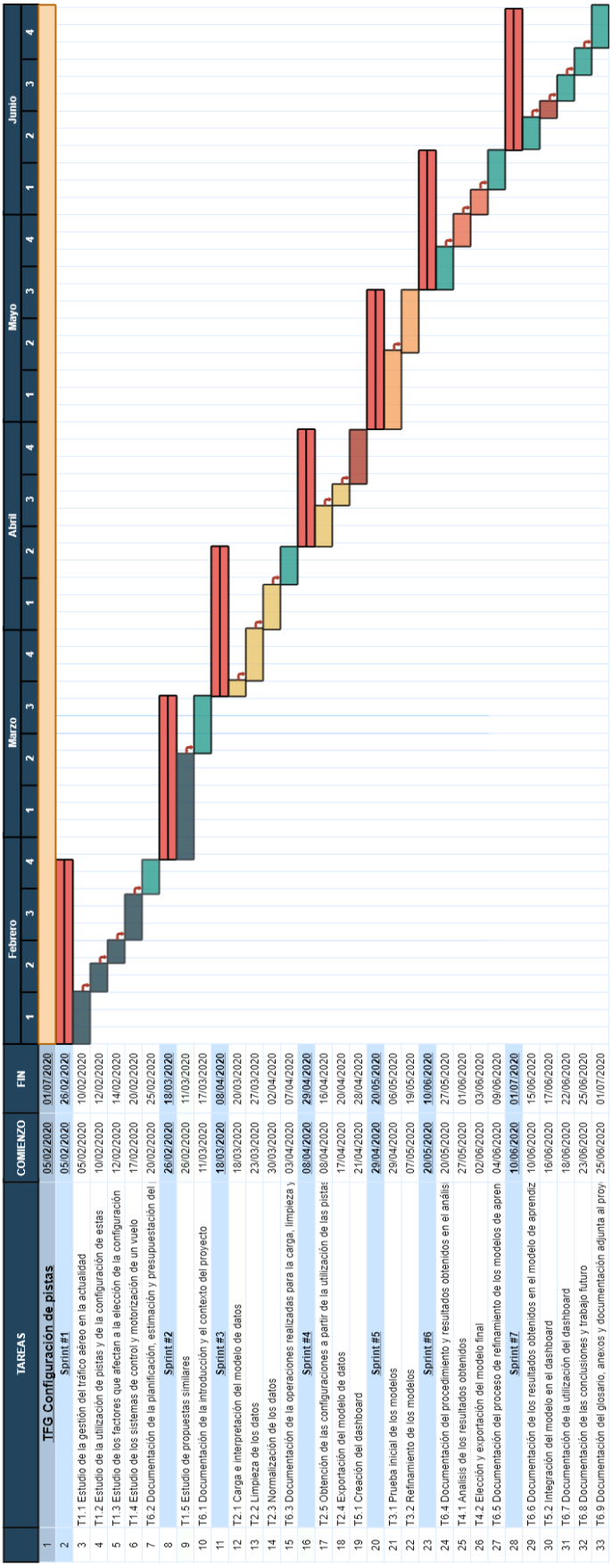
3.3.2. Planificación económica

Debido a los ajustes en la planificación llevados a cabo, es necesario actualizar los costes asociados al proyecto que se han visto afectados. A pesar de que se haya producido ajustes, la planificación temporal no se ha visto afectada, por lo que los costes asociados a los recursos humanos no han sufrido variaciones. Por contra, la introducción de una nueva tarea, ha supuesto un incremento en el uso de los recursos hardware, cuyos costes han sido actualizados tal y como se muestra en la Tabla 3.12.

Por tanto el coste real del proyecto ha sido de **4.662,25€**, frente a los 4.629,93€ que se estimaron inicialmente, sufriendo un **sobrecoste de 32,32€**.

Herramienta	**Coste (mes)**	**Vida útil**	**% de uso**	**Total**
Ordenador Personal	800€ (coste total)	5 años	8,33 %	66,64€
Internet	20€	5 meses	100 %	100€
Coste Energético	7,2€	5 meses	100 %	37,5€
Máquina Virtual	64,64€	3,5 meses	70 %	226,24€
GitLab	0€	–	–	0€
Visual Studio Code	0€	–	–	0€
Overleaf	0€	–	–	0€
Anaconda	0€	–	–	0€
Trello	0€	–	–	0€
Slack	0€	–	–	0€
TOTAL				**430,38€**

Tabla 3.12: Costes actualizados asociados a herramientas y recursos hardware.



	TAREAS	COMIENZO	FIN
1	**TFG Configuración de pistas**	05/02/2020	01/07/2020
2	**Sprint #1**	05/02/2020	26/02/2020
3	T1.1 Estudio de la gestión del tráfico aéreo en la actualidad	05/02/2020	10/02/2020
4	T1.2 Estudio de la utilización de pistas y de la configuración de estas	10/02/2020	12/02/2020
5	T1.3 Estudio de los factores que afectan a la elección de la configuración	12/02/2020	14/02/2020
6	T1.4 Estudio de los sistemas de control y motorización de un vuelo	17/02/2020	20/02/2020
7	T6.2 Documentación de la planificación, estimación y presupuestación del	20/02/2020	25/02/2020
8	**Sprint #2**	26/02/2020	18/03/2020
9	T1.5 Estudio de propuestas similares	26/02/2020	11/03/2020
10	T6.1 Documentación de la introducción y el contexto del proyecto	11/03/2020	17/03/2020
11	**Sprint #3**	18/03/2020	08/04/2020
12	T2.1 Carga e interpretación del modelo de datos	18/03/2020	20/03/2020
13	T2.2 Limpieza de los datos	23/03/2020	27/03/2020
14	T2.3 Normalización de los datos	30/03/2020	02/04/2020
15	T6.3 Documentación de la operaciones realizadas para la carga, limpieza y	03/04/2020	07/04/2020
16	**Sprint #4**	08/04/2020	29/04/2020
17	T2.5 Obtención de las configuraciones a partir de la utilización de las pistas	08/04/2020	16/04/2020
18	T2.4 Exportación del modelo de datos	17/04/2020	20/04/2020
19	T5.1 Creación del dashboard	21/04/2020	28/04/2020
20	**Sprint #5**	29/04/2020	20/05/2020
21	T3.1 Prueba inicial de los modelos	29/04/2020	06/05/2020
22	T3.2 Refinamiento de los modelos	07/05/2020	19/05/2020
23	**Sprint #6**	20/05/2020	10/06/2020
24	T6.4 Documentación del procedimiento y resultados obtenidos en el análisi	20/05/2020	27/05/2020
25	T4.1 Analisis de los resultados obtenidos	27/05/2020	01/06/2020
26	T4.2 Elección y exportación del modelo final	02/06/2020	03/06/2020
27	T6.5 Documentación del proceso de refinamiento de los modelos de apren	04/06/2020	09/06/2020
28	**Sprint #7**	10/06/2020	01/07/2020
29	T6.6 Documentación de los resultados obtenidos en el modelo de aprendiz	10/06/2020	15/06/2020
30	T5.2 Integración del modelo en el dashboard	16/06/2020	17/06/2020
31	T6.7 Documentación de la utilización del dashboard	18/06/2020	22/06/2020
32	T6.8 Documentación de las conclusiones y trabajo futuro	23/06/2020	25/06/2020
33	T6.9 Documentación del glosario, anexos y documentación adjunta al proy	25/06/2020	01/07/2020

Figura 3.2: Diagrama de Gantt final del proyecto.

Capítulo 4

Estudio del dominio

4.1. Gestión del tráfico aéreo

A pesar de que la tarea de gestionar el tráfico aéreo se viene realizando desde hace bastantes años, la gestión del tráfico aéreo (*Air Traffic Management*, ATM, en inglés), se convierte en una tarea más importante, compleja e imprescindible. Esto se debe a que en los últimos años, el número de vuelos realizados ha sufrido un crecimiento exponencial que además, se espera que aumente a lo largo de los próximos años [10], haciendo aún mas necesaria la tarea de gestionar y controlar ese gran volumen de operaciones. Además de esto, los sistemas de gestión del tráfico aéreo se desarrollan en entornos complejos en los que coexisten aeronaves, aeropuertos o aerolíneas, cuyas prioridades se encuentran sujetas a factores externos. Alguno de estos factores son: la meteorología, el volumen de tráfico aéreo, la capacidad y estructura del aeropuerto o la diversidad en cuanto a los protocolos y procedimientos de control que se aplican. Todos estos factores hacen que la labor de la gestión del tráfico aéreo sea muy importante y a la vez compleja.

Para llevar a cabo la labor de gestionar el tráfico aéreo, es necesaria la instalación de nuevos sistemas y tecnologías que permitan una comunicación más eficaz y eficiente entre los aeropuertos y las aeronaves (además de entre las propias aeronaves), consiguiendo así mejorar tanto la transmisión de los datos entre las entidades implicadas como la administración de los mismos. Las principales metas que se pretenden conseguir con esta mejora en los sistemas de gestión del tráfico aéreo son:

1) Mejorar y garantizar la seguridad de las aeronaves durante un vuelo, evitando las colisiones entre estas, tanto a lo largo del vuelo como a la hora de realizar las maniobras en las proximidades del aeródromo.

2) Prevenir obstrucciones en las áreas próximas al aeropuerto, evitando así la acumulación de aeronaves y disminuyendo la congestión del tráfico en el aeropuerto.

3) Agilizar y mantener un flujo ordenado de las operaciones que se llevan a cabo.

4) Proporcionar tanto a las aeronaves como al aeropuerto asesoramiento e información útil para la conducción segura y eficiente de los vuelos.

5) Optimizar las operaciones de vuelo que se llevan a cabo para disminuir la contaminación generada a causa de las mismas.

En la actualidad, existen diferentes iniciativas con el objetivo de mejorar la gestión del tráfico aéreo. Las más importantes y sobre las que trataremos en la presente memoria son: *Single European Sky ATM Research* (SESAR), enfocada en el espacio aéreo europeo, y *Next Generation Air Transportation System* (NextGen), enfocado al espacio aéreo estadounidense.

4.2. SESAR

Con el objetivo de llevar a cabo la constitución de un espacio aéreo único europeo (*Single European Sky*), nace el proyecto SESAR, un proyecto colaborativo llevado a cabo por la Unión Europea con el propósito de constituir y poner en práctica una política común en cuanto a la gestión del tráfico aéreo en todo el espacio aéreo europeo. Esta propuesta tecnológica es una de las más innovadoras llevadas a cabo en Europa, ya que es necesario efectuar una gran modificación sobre los actuales sistemas de control y comunicación, tanto en los aeropuertos como en las aeronaves.

El principal objetivo de este proyecto consiste en implantar en el año 2020 una red ATM que permita mejorar tanto el rendimiento como la eficiencia de la actual red de tráfico aéreo, permitiendo así gestionar un mayor número de operaciones y hacer frente a este aumento en el número de vuelos. Se estima que con este proyecto se podrá gestionar hasta tres veces más operaciones que con el sistema actual, disminuir el impacto medioambiental en aproximadamente un 10-15 %, mejorar la seguridad de las operaciones que se realicen, disminuir los tiempos de espera causados por una gestión ineficiente del tráfico aéreo, disminuir los costes asociados a cada vuelo, etc.

La visión actual del proyecto SESAR se basa en los Proveedores de Servicios de Navegación Aérea (*Air Navigation Service Provider*, ANSP, en inglés) de tal forma que los aviones puedan realizar la trayectoria correspondiente al vuelo, sin ningún tipo de limitación asociada a la configuración del tráfico aéreo (ver Figura 4.1). Esta hazaña se consigue gracias al aumento progresivo de tecnologías de virtualización y de automatización, permitiendo a los proveedores de servicios de navegación aérea realizar operaciones donde sea necesario, debido a la compatibilidad de los aeropuertos con esta nueva red de ATM, facilitando y optimizando las operaciones de los usuarios del espacio aéreo. El proyecto SESAR pretende alcanzar entre los años 2035 y 2050 que este tipo de operaciones se realicen en toda Europa, consiguiendo una coordinación completa entre los servicios de navegación aérea[1].

4.3. NextGen

The Next Generation Air Transportation System[2] (NextGen), es un proyecto liderado por la Administración Federal de la Aviación (*Federal Aviation Administration*, en inglés), cuyo principal objetivo consiste en la modernización del espacio aéreo estadounidense, mejorando la seguridad, eficiencia y costes de los vuelos. Al igual que ocurre con el proyecto (SESAR), NextGen es uno de los proyectos más ambiciosos de la historia de los Estados Unidos y se espera que las mejoras introducidas a las tecnologías de gestión del tráfico aéreo actuales permitan gestionar a más de 2,7 millones de pasajeros diarios y más de 44.000 vuelos diarios (ver Figura 4.2).

El funcionamiento de NextGen se centra en los siguientes aspectos:

[1] https://www.airport-suppliers.com/suppliers/air-navigation-service-provider/.
[2] https://www.faa.gov/nextgen/what_is_nextgen/.

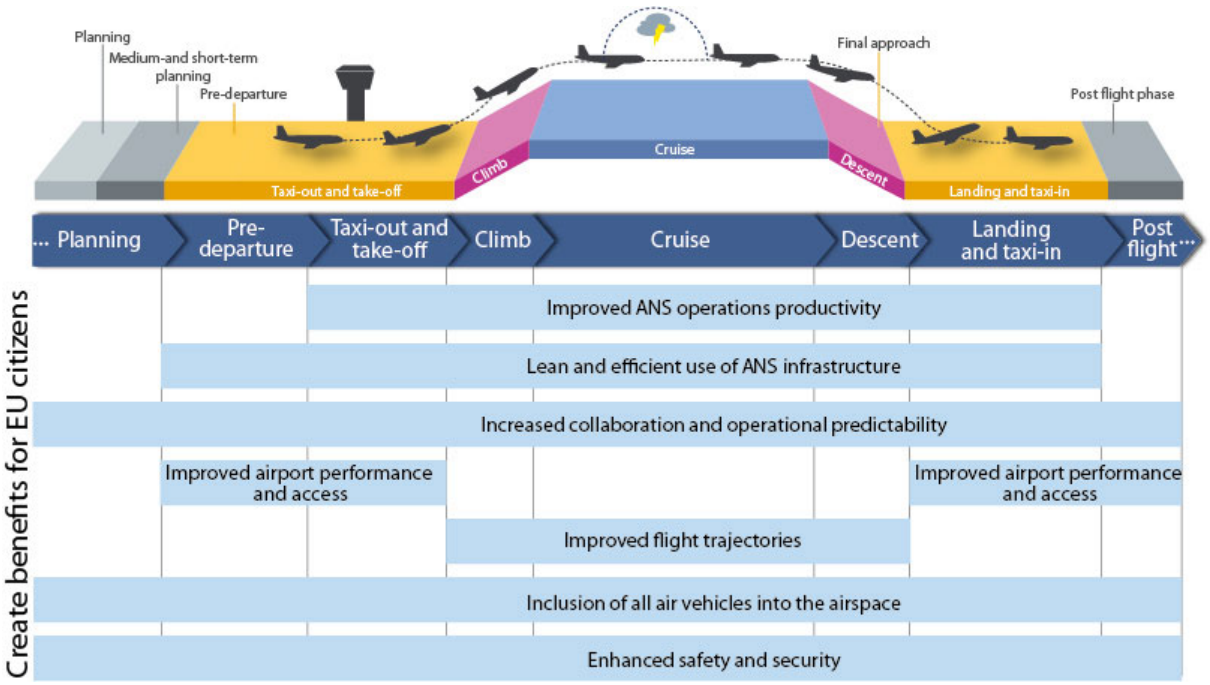


Figura 4.1: Esquema objetivos del proyecto SESAR Fuente: *https://www.sesarju.eu/vision (visitado 14-03-2020).*

1) **Vigilancia:** La implementación de NextGen proporciona a los controladores la ubicación exacta de la aeronave, así como una visión clara tanto del entorno meteorológico como de la propia aeronave.

2) **Navegación:** NextGen sustituye las tradicionales ayudas de navegación basadas en tierra por un sistema de navegación por satélite. Esto permite la creación de rutas óptimas, reducir los tiempos tanto durante un vuelo como los de espera, además de disminuir el uso de combustible por parte de la aeronave, reduciendo el gasto asociado a cada vuelo.

3) **Comunicación:** Las aeronaves deben estar capacitadas para poder enviar y recibir instrucciones dinámicas y complejas de los sistemas terrestres que permitan identificar dónde se encuentran, a qué hora y en qué condiciones. De esta forma se consigue una comunicación más rápida, eficiente y sin las limitaciones que sufrían los sistemas de radio tradicionales.

4.4. Sistemas de control ADS

Los sistemas de control ADS (Vigilancia Dependiente Automática, *Automatic Dependent Surveillance*, en inglés)[3] constituyen una tecnología para el control del tráfico aéreo que permite que todas las aeronaves que dispongan de esta tecnología envíen de forma automática, mediante un enlace de datos, la información de los sistemas de navegación y de determinación de la posición de los que la aeronave dispone a bordo [15].

Los sistemas de control ADS tienen las siguientes características:

[3] http://www.ads-b.com/.

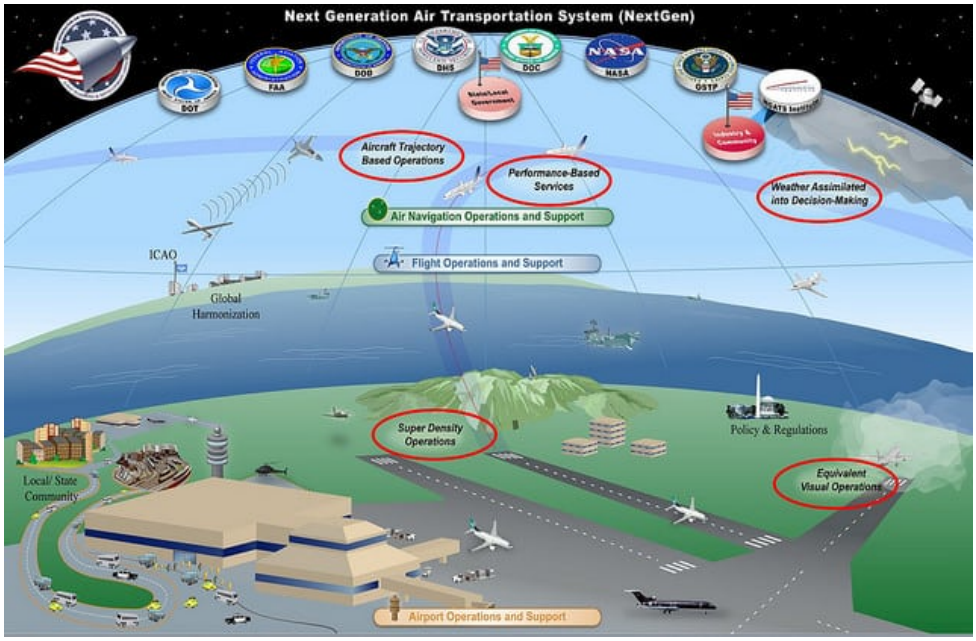


Figura 4.2: Estructura NextGen Fuente: *https://www.faa.gov/nextgen/what_is_nextgen/ (visitado 12-03-2020).*

1) **Autónomo:** No necesita de la intervención de ningún operador para efectuar el envío de los datos de la aeronave.

2) **Dependiente:** Los datos enviados a los distintos receptores (torres de control u otras aeronaves cercanas) dependen por completo de las capacidades de equipamiento de la aeronave de origen del mensaje ADS.

3) **Semejanza de datos:** Los datos proporcionados son similares a los datos del radar tanto para los controladores de tierra como para el resto de las aeronaves.

En la actualidad podemos encontrar dos versiones de ADS: ADS-B (Vigilancia Dependiente Automática-Difusión, *Automatic Dependent Surveillance-Broadcast*, en inglés) y ADS-C (Vigilancia Dependiente Automática-Contrato, *Automatic Dependent Surveillance-Contract*, en inglés).

- **ADS-B:** Utiliza los principios de ADS para transmitir de forma automática y periódica información de posición de la aeronave: identificador, vector de velocidad, altitud, estado del vuelo etc (ver Figura 4.3). Esta información se manda de forma abierta a todos los receptores que se encuentren dentro del radio de influencia de los mensajes ADS-B (distancia máxima desde la que se puede capturar un mensaje ADS-B). Esta versión se ha convertido en la principal aplicación de ADS en la actualidad.

  Además, dentro de ADS-B existen dos tipos de componentes:

  - **ADS-B Out:** Se encarga de enviar los datos de la aeronave (posición, altitud, velocidad, estado, etc).
  - **ADS-B In:** Recibe información relevante para el piloto: datos de otras aeronaves, previsiones meteorológicas, restricciones de vuelo, etc.

- **ADS-C:** Su funcionamiento es muy similar al utilizado en ADS-B con la diferencia de que, en este caso, los datos se transmiten entre una aeronave y un ANSP previo contrato explícito entre ambas. Este contrato representa las condiciones por las cuales la aeronave ha de enviar el mensaje ADS al ANSP y pueden ser por demanda (el ANSP solicita a la aeronave el mensaje), periódico (el mensaje se envía de manera periódica), de evento (cuando ocurre alguna circunstancia concreta) y/o de emergencia (cuando ocurre una situación de emergencia). Esta versión de ADS se utiliza con mayor frecuencia en vuelos en áreas transoceánicas o transcontinentales en los que los niveles de tráfico aéreo son relativamente bajos.

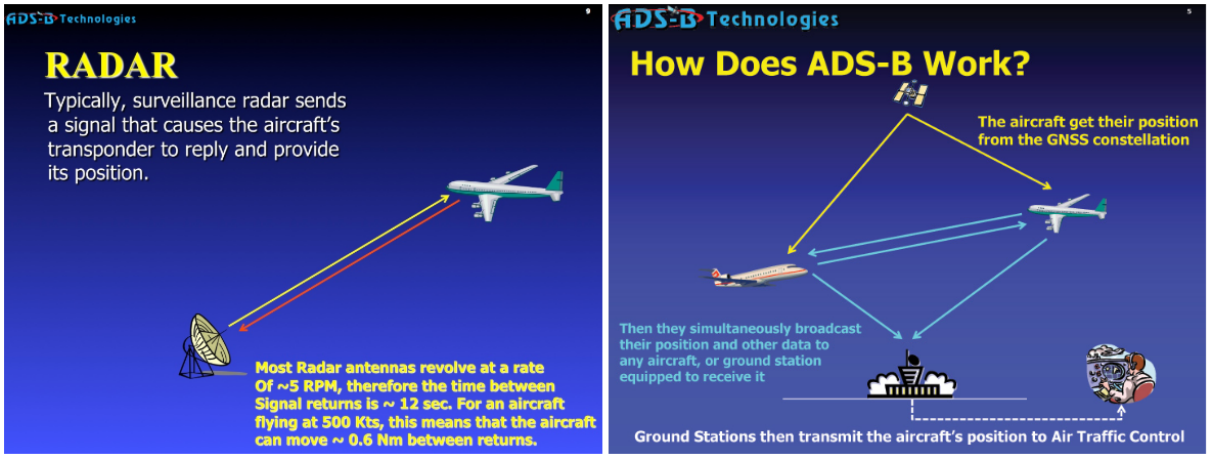


Figura 4.3: Funcionamiento sistema ADS-B Fuente: *http://www.ads-b.com/ (visitado 14-03-2020).*

Este tipo de tecnologías facilitan las tareas de gestión de tráfico aéreo debido a que permiten en todo momento conocer el estado de las aeronaves gracias al continuo envío de mensajes ADS-B por parte de estas, consiguiendo una mejor eficiencia y seguridad a la hora de tomar decisiones de gestión. De igual forma, la tecnología ADS permite a las aeronaves conocer la situación de otras aeronaves cercanas sin necesidad de que exista una torre de control en las proximidades, ya que estos mensajes pueden ser recibidos entre aeronaves directamente, haciendo posible la transmisión de datos en lugares donde no haya torres de control cercanas, por ejemplo, en vuelos transoceánicos.

Algunas de las ventajas que presentan este tipo de sistemas son:

- Conocer en tiempo real el estado de las aeronaves próximas, facilitando así la gestión del tráfico aéreo, reduciendo los tiempos de espera y mejorando la seguridad global de las operaciones aéreas que se realizan.
- Posibilidad de reconstrucción de la trayectoria de vuelo de una aeronave [21].
- Identificación de estado y posición de las aeronaves próximas en lugares donde no existen torres control cercanas.
- El piloto de la aeronave tiene disponible en todo momento información de las aeronaves que lo rodean y viceversa. Esta información incluye datos de posicionamiento de las aeronaves (altitud, velocidad, rumbo) y su estado.

- Reducción del uso del canal de comunicación destinado a la transmisión de mensajes.

Finalmente, en cuanto a los inconvenientes que presenta esta tecnología, el principal de todos ellos es la escalabilidad, ya que cada aeronave envía un mensaje ADS-B cada 0,5 segundos. Esto implica que en zonas con una gran confluencia de aeronaves, el volumen de datos que se genera sea difícil de gestionar por parte de los receptores, ya que cada segundo recibe dos mensajes ADS-B de todas las aeronaves que se encuentren dentro de su radio de influencia.

4.5. Pistas

Dentro de la infraestructura de un aeropuerto, una pista es una superficie rectangular destinada a la realización de las maniobras de aterrizaje y despegue por parte de las aeronaves. Como se puede ver en la Sección 4.8, uno de los factores que influyen a la hora de asignar una pista a una aeronave es la meteorología, concretamente a la dirección y fuerza del aire. Para solventar este problema, la mayoría de aeropuertos cuentan con varias pistas, situadas en diferentes ubicaciones y con distintas orientaciones dentro del aeropuerto. Dentro de un aeropuerto podemos distinguir dos tipos de pistas:

- **Pista Principal:** La pista principal es la pista que se utiliza de forma usual en el aeropuerto siempre y cuando las condiciones para su uso sean adecuadas (meteorología, estado de la pista, tráfico aéreo etc).

- **Pista Secundaria:** La pista secundaria es una pista auxiliar que se utiliza cuando la pista principal no se encuentre disponible ya sea por motivos meteorológicos o por motivos logísticos. Su función es servir como apoyo a la pista principal, consiguiendo así que las operaciones dentro del aeropuerto no se vean afectadas al no poder utilizar la pista principal.

El Anexo 14 al Convenio sobre Aviación Civil Internacional, Capítulo 3 [20], indica las características físicas que han de cumplir las pistas en los aeropuertos ya sean principales o secundarias. Entre estas características, las más influyentes a la hora de gestionar la configuración de las pistas (ver Figura 4.5) son las siguientes:

1) **Número y orientación:** Tanto el número como la orientación de las pistas viene determinado por el *coeficiente de utilización*. Este coeficiente representa el porcentaje de tiempo en el que la pista no se ve afectada por vientos transversales, siendo recomendable que este sea superior al 95 %.

   El cálculo de la orientación de las pistas es uno de los puntos más importantes a la hora de su creación, ya que en ellas es donde se realizarán las maniobras más delicadas a lo largo de un vuelo. Una de las técnicas que se utilizan para el análisis y la toma de esta decisión es la representación de la velocidad del aire mediante la rosa de los vientos. Este diagrama consta de una circunferencia en la que se encuentran representados los rumbos posibles junto con la fuerza del viento en esa dirección (ver Figura 4.4).

2) **Longitud:** La longitud de las pistas no tiene un valor fijo establecido, sino que dependerá de varios factores tales como: las características físicas de las aeronaves que la utilizarán (peso, longitud, anchura), características del aeropuerto y de la pista (altitud del aeropuerto,

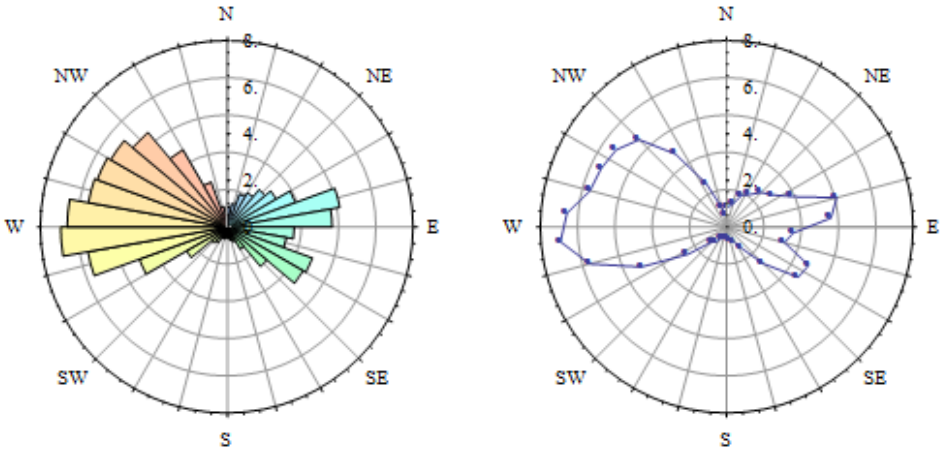


Figura 4.4: Representación utilizando la técnica de la rosa de los vientos, Fuente: *Wechoosethemoon (visitado 16-03-2020).*

pendiente de la pista) y características meteorológicas (humedad, temperatura máxima registrada en el aeropuerto). La combinación de estos agentes proporciona la longitud mínima que ha de tener una pista para que un determinado avión pueda utilizarla (generalmente se toman valores por exceso para así suplir posibles errores en las mediciones y aumentar la seguridad). Estas pistas solo podrán ser utilizadas por los aviones cuyas características técnicas sean iguales o inferiores a las utilizadas para el cálculo de la longitud.

3) **Anchura:** La anchura en las pistas dispone de unas medidas base establecidas dependiendo del código de referencia aeroportuario de la aeronave[4]. Este código (compuesto por un número y una letra) sirve para clasificar cada aparato de tal forma que el número representa la longitud mínima necesaria para el despegue en condiciones normales (sin pendiente en la pista, ni viento) y la letra representa una combinación entre la envergadura de la aeronave y la distancia que existe entre el tren de aterrizaje superior e inferior. Cada aeronave se encuentra representada por un código de referencia de acuerdo a la Tabla 4.1.

Número	**Longitud mínima necesaria**
1	hasta 800m
2	de 800m a 1200m
3	de 1200m a 1800m
4	más de 1800m

Letra	**Envergadura**	**Distancia externa del tren de aterrizaje**
A	hasta 15 m	hasta 4,5 m
B	de 15m a 24m	de 4,5m a 6m
C	de 24m a 36m	de 6m a 9m
D	de 36m a 52m	de 9m a 14m
E	de 52m a 65m	de 9m a 14m
F	de 65m a 80m	de 14m a 16m

Tabla 4.1: Referencia para la asignación del código de referencia aeroportuario, Fuente: [29], [20].

[4]https://www.skybrary.aero/index.php/ICAO_Aerodrome_Reference_Code.

La anchura mínima de una pista para que una determinada aeronave pueda utilizarla viene definida por el código de referencia del aparato de acuerdo a la siguiente tabla:

	Letra					
Número	**A**	**B**	**C**	**D**	**E**	**F**
1	18m	18m	23m	-	-	-
2	23m	23m	30m	-	-	-
3	30m	30m	30m	45m	-	-
4	-	-	45m	45m	45m	60m

Tabla 4.2: Anchura mínima de la pista para cada tipo de aeronave. Fuente: [20].

4) **Señalización:** La señalización es otro de los factores importantes a la hora de la gestión de las pistas, siendo esta la forma que tiene el piloto de localizar la pista en la que se va a efectuar las maniobras así como la de obtener las características referentes a esa pista (localización, orientación, anchura etc). Algunas de las señales más importantes son las siguientes:

   a) **Número en la pista de aterrizaje:** Debido a que las pistas pueden ser utilizadas en cualquiera de los dos sentidos (dependiendo de las condiciones climatológicas), es necesaria la existencia de una señal para distinguir ambas direcciones.

      La forma en la que se representa estas direcciones es con un número (generalmente de dos cifras), situado en el inicio de ambos extremos de la pista. Este número representa la orientación de la pista con respecto del eje magnético (redondeado en decagrados), por tanto si un extremo de la pista se encuentra representado con el valor 11 (110 grados), su homólogo en el otro extremo será 29 (290 grados) ($110 + 180 = 290 \rightarrow 29$). Además de esto, este número se utiliza también para identificar la pista.

      Este tipo de nomenclatura presenta un claro problema ya que si un aeropuerto dispone de más de un par de configuraciones de pistas paralelas (ver Sección 4.6) con la misma orientación, el número identificativo será el mismo en ambas. Para solucionar este inconveniente y evitar confusiones, se varía en una unidad el número de uno de los pares de pistas, de tal forma que si hay dos pistas representadas con el valor 11, una de ellas se deja con ese valor y la otra pasa a estar representada con el valor 10 o con el 12 (valores contiguos a la pista con orientación repetida).

   b) **Letra:** En el caso de que el aeropuerto disponga de varias pistas situadas en paralelo (ver Sección 4.6), además del número que indica la orientación, se añade también una letra para ayudar a distinguirlas. Estas letras son: L (en el caso de que la pista sea la de la izquierda), C (para la del centro) y R (para la de la derecha). Por ejemplo:

      - Para dos pistas paralelas: “L” “R”.
      - Para tres pistas paralelas: “L” “C” “R”.
      - Para cuatro pistas paralelas: “L” “R” “L” “R”.
      - Para cinco pistas paralelas: “L” “C” “R” “L” “R” o “L” “R” “L” “C” “R”.
      - Para seis pistas paralelas: “L” “C” “R” “L” “C” “R”.

   c) **Señal de ancho de pista:** Están constituidas por franjas dispuestas horizontalmente una al lado de otra (como un paso de peatones) al inicio de ambos extremos de la pista (antes

de la señalización de la orientación). Sirven para indicar el ancho total de la pista. La relación entre el número de franjas y la anchura es la siguiente:

Número de franjas	**Ancho de la pista**
4 franjas	18m
6 franjas	23m
8 franjas	30m
12 franjas	45m
16 franjas	60m

Tabla 4.3: Relación entre el número de franjas y la anchura de la pista. Fuente: *sociedadaeronautica.org (visitado 17-03-2020).*

Figura 4.5: Ejemplo de señalización en una pista de aeropuerto, Fuente: *Diario del Viajero (visitado 15-03-2020).*

4.6. Configuración de las pistas

La configuración de las pistas (*runway configuration* en inglés), hace referencia al número, orientación y agrupamiento de las pistas dentro de un aeropuerto. Existen varios tipos de configuraciones aunque muchos de ellos se encuentran constituidos por la combinación de otros (ver Figura 4.6). Los principales tipos de configuración son los siguientes:

1) **Pista simple:** Esta es la configuración de pistas más básica que se puede realizar en un aeropuerto. Consta de una única pista en la que se llevan a cabo las operaciones de aterrizaje o de despegue.

2) **Pistas paralelas:** Las pistas paralelas están constituidas por la combinación de dos o más pistas simples. Una de las características de este tipo de configuración es que las pistas dis-

ponen de la misma orientación, aumentando así el número de aeronaves que pueden utilizarla y por consiguiente, reducir la congestión en el tráfico aéreo.

3) **Pistas que se cortan:** Las pistas que se cortan son un tipo de configuración en la que se combinan dos o más pistas simples de manera que existe una intersección entre ambas. Este tipo de configuración se implementa cuando existen vientos fuertes en más de una dirección, haciendo inseguro el uso de la pista en una única dirección. Gracias a ello se consigue un aumento en la versatilidad del aeropuerto al disponer de varias pistas con distinta orientación y utilizar la que permita la realización de las maniobras de una forma más segura. No obstante, se ha de evitar en la medida de lo posible este tipo de configuración ya que, al existir un punto de intersección entre ambas, implica que en un determinado punto la aeronave utiliza ambas pistas, impidiendo el uso de la otra pista por otras aeronaves, por lo que es preferible tenerlas de forma independiente, evitando así problemas a la hora de gestionarlas.

4) **Pistas en V abierta:** Las pistas en V abierta son el último tipo de configuración de pistas. Están constituidas por dos pistas simples en direcciones opuestas que no llegan a cortarse nunca. Al igual que las pistas que se cortan, las pistas en V se utilizan principalmente cuando los vientos soplan con intensidad en varias direcciones y no es segura la utilización de la pista en una única dirección. La principal diferencia con respecto a las pistas que se cortan es que las pistas en V no tienen intersección en ningún punto del recorrido, haciendo más sencilla su gestión ya que en ningún momento se llega a "invadir" otra pista (como si ocurre en la intersección entre las pistas que se cortan).

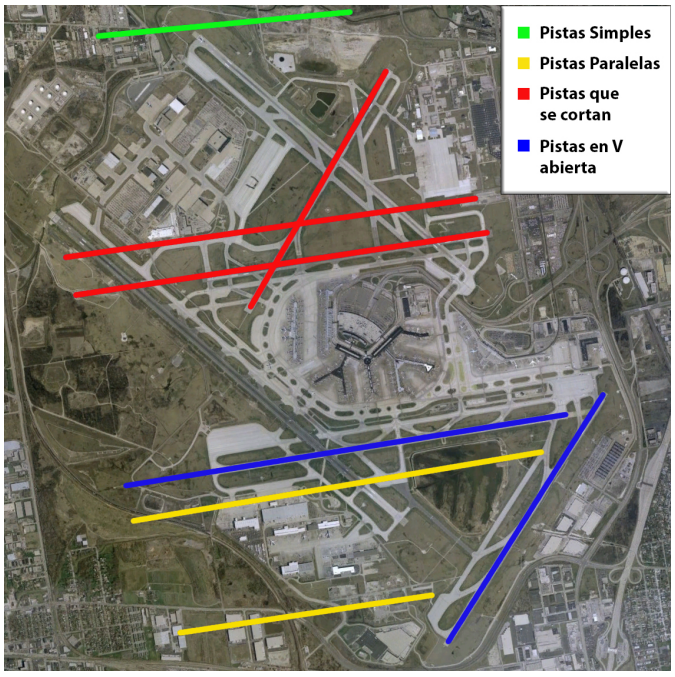


Figura 4.6: Ejemplo configuración de pistas en un aeropuerto.

4.7. Capacidad operativa

La capacidad operativa es la cantidad de operaciones que una pista es capaz de albergar en un espacio de tiempo. Generalmente, se suele representar como la cantidad de operaciones que se realizan en una pista durante una hora (Op/h a partir de ahora).

Esta medida es muy relevante a la hora de determinar qué configuración de pistas elegir puesto que una capacidad operativa baja sobre una pista con una alta demanda de utilización puede ralentizar el tráfico aéreo en todo el aeropuerto, junto con los retrasos que esto supondría. Es por ello, por lo que se ha de establecer un equilibrio entre la seguridad (elegir aquellas configuraciones en las que se minimice el riesgo a la hora de realizar las maniobras) y la productividad (maximizar el número de operaciones que se pueden realizar por hora en cada pista).

Para realizar la estimación de la capacidad operativa en una configuración de pistas tenemos que distinguir dos tipos de condiciones:

- **VFR:** También conocidas como Reglas de Vuelo Visual (*Visual Flight Rules*, en inglés), comprenden un conjunto de normas recogidas en el Reglamento de Circulación Aérea [7] en las que se establecen las condiciones por las cuales el piloto puede manejar la aeronave sin la necesidad de utilizar los aparatos de visión asistida. Esto se debe a que el piloto dispone en todo momento de un contacto visual con los elementos del terreno, reduciendo así la necesidad de recurrir a ayuda suplementaria. Como consecuencia, el tiempo empleado en la realización de las maniobras es menor que si no dispusiera de una visión del entorno, aumentando la capacidad operativa de las pistas implicadas.

- **IFR:** Conocidas como Reglas de Vuelo Instrumental (*Instrument Fight Rules*, en inglés). Al igual que las VFR, son un conjunto de normas que se recogen en el Reglamento de Circulación Aérea en las que se regula el vuelo de aeronaves utilizando aparatos de ayuda asistida. Esto implica que en los casos en los que se utilicen estas normas, el piloto no necesariamente ha de tener un contacto visual con el entorno, haciendo más lenta la realización de las maniobras y reduciendo la capacidad operativa de las pistas implicadas.

Por estas razones, la configuración de las pistas elegidas junto con las condiciones meteorológicas (ver Sección 4.8), determinarán la capacidad operativa del aeropuerto. Según [13] la capacidad operativa aproximada para cada una de las configuraciones mencionadas en la Sección 4.6 es la siguiente:

1) **Pistas Simples:** En una configuración de pistas simples, se estima que la capacidad operativa en condiciones VFR (el piloto dispone de contacto visual con el entorno) ronda entre las 50 y 100 Op/h. Por otro lado, si el piloto no dispone de contacto visual con el entorno (condiciones IFR), la capacidad operativa disminuye y se sitúa en torno a las 50-70 Op/h, dependiendo de las ayudas asistidas a la navegación que disponga la aeronave.

2) **Pistas paralelas:** La capacidad operativa en una configuración de pistas paralelas dependerá en gran medida de la separación que exista entre las pistas que la conforman, siendo menor cuanto más próximas se encuentren. Para condiciones VFR, la capacidad operativa no se ve afectada por la separación que existe entre las pistas (salvo que estén operando aeronaves de grandes dimensiones) y oscila entre las 100-200 Op/h.

Por otro lado, para condiciones IFR, la capacidad operativa sí que se ve afectada por la distancia entre las pistas, siendo de 50-60 Op/h para pistas con poca separación; 70-80 Op/h para pistas con una separación intermedia y de 85-105 Op/h para pistas con una gran separación.

3) **Pistas que se cortan:** En una configuración de pistas que se cortan, la capacidad operativa depende de la zona de intersección entre las ellas, de tal forma que cuanto más alejado se encuentre el punto de intersección de los extremos de las pistas que conforman esta configuración, menor será su capacidad operativa. Del mismo modo, la capacidad operativa será mayor si la intersección entre las pistas se encuentra en los extremos de la pistas implicadas. La capacidad operativa máxima en este tipo de configuración es de 70-175 Op/h en condiciones VFR y de 60-70 Op/h en condiciones IFR.

4) **Pistas en V abierta:** En una configuración de pistas en V abierta, la capacidad operativa depende de la dirección en la que se realicen las maniobras, siendo menor si estas se realizan hacia el vértice de la V. Para condiciones VFR en las que las operaciones se realizan hacia fuera del vértice, la capacidad operativa es de 80-200 Op/h y de 50-100 Op/h si las operaciones se realizan hacia el vértice. Del mismo modo, para condiciones IFR en las que las operaciones se realizan hacia fuera del vértice, la capacidad operativa es de 60-70 Op/h, mientras que si las operaciones se realizan hacia el vértice, la capacidad operativa disminuye hasta las 50-60 Op/h.

En la Tabla 4.4 se resume la capacidad operativa de cada una de las configuraciones en función de las condiciones visuales del piloto y de las características de las pistas:

Configuración / Condiciones	**Pistas Simples**	**Pistas Paralelas**			**Pistas que se cortan**	**Pistas en V abierta**	
		Próximas	**Separación Intermedia**	**Mucha Separación**		**Hacia el vértice**	**Hacia fuera del vértice**
VFR	50-100 Op/h	100-200 Op/h			70-175 Op/h	50-100 Op/h	80-200 Op/h
IFR	50-70 Op/h	50-60 Op/h	70-80 Op/h	85-105 Op/h	60-70 Op/h	50-60 Op/h	60-70 Op/h

Tabla 4.4: Capacidad Operativa para cada configuración de pista en función de las condiciones visuales.

4.8. Meteorología

La meteorología es uno de los factores más influyentes a la hora de establecer las configuraciones de las pistas en los aeropuertos, siendo esta la responsable del 69 % de los retrasos del tráfico aéreo en el Sistema Nacional del Espacio Aéreo [17]. Actualmente, la mayoría de los grandes aeropuertos disponen de su propia estación meteorológica de la cual obtienen datos sobre el clima que posteriormente utilizarán tanto para llevar un histórico, como para realizar predicciones meteorológicas en las inmediaciones del propio aeropuerto. Esto supone un gran avance para la gestión del tráfico aéreo, ya que la capacidad de pronosticar posibles factores climáticos adversos con anterioridad es crucial para poder generar un plan alternativo en el que se consiga mitigar o evitar por completo el impacto debido al clima.

Dentro de los factores atmosféricos que más afectan a la seguridad hay que destacar las ráfagas de viento, la niebla y las tormentas. Por un lado, las ráfagas de viento pueden llegar a generar turbulencias, dificultando al piloto el manejo de la aeronave, sobre todo durante las

maniobras de aproximación, ascenso, descenso, aterrizaje y despegue. Del mismo modo, durante las tormentas, además de las ráfagas de viento, se suma el problema de los rayos, granizo o heladas. Estas condiciones son muy peligrosas ya que las heladas pueden producir una capa de hielo en la estructura de la aeronave haciendo que su peso y aerodinámica varíe[5].

Además del problema que supone la creación de la capa de hielo en la estructura de la aeronave, se une también el problema del granizo, ya que el golpeo constante de este en la estructura del avión puede hacer que el piloto llegue a perder el control.

Figura 4.7: Factores meteorológicos que afectan a las operaciones del aeropuerto. Fuente: [17].

Es importante destacar el impacto que suponen las tormentas eléctricas durante un vuelo ya que, según un estudio realizado a profesionales de la aviación, el 75 % de ellos seleccionaron las tormentas como principal factor meteorológico que afecta a sus operaciones [17]. Esto se debe a que las tormentas eléctricas severas generalmente incluyen muchas de las amenazas vistas en la Figura 4.7, como las ráfagas de viento, granizo, lluvia o relámpagos. Además de esto, cabe destacar que el 80 % de los rayos que se producen durante una tormenta son de tipo nube a nube, es decir, el rayo "salta" de una nube a otra haciendo que sea muy peligroso volar en las proximidades y en el interior de una tormenta eléctrica.

Por otra parte, además de las tormentas y el viento, las nubes suponen un gran problema durante el vuelo. Concretamente, el tipo de nube que presenta mayor riesgo son las *cumulonimbus* (ver Figura 4.8). Este tipo de nube se forma combinando masas de aire de distinta temperatura y humedad con fuertes corrientes de aire caliente y frío que ascienden y descienden dentro de la nube. Como consecuencia, tanto en el interior como en las proximidades se pueden generan turbulencias, tormentas eléctricas, granizo o heladas. Es por ello por lo que durante el transcurso de un vuelo, se evita el acercamiento a ellas todo lo posible[6].

[5] https://www.bbc.com/mundo/noticias/2014/12/141229_como_mal_tiempo_afecta_aviones_indonesia_mes.
[6] https://www.miedoalosaviones.com/cumulonimbu-peligrosas-los-aviones/.

Figura 4.8: Nube cumulonimbus, Fuente: *The Weather Channel (visitado 17-03-2020).*

No obstante, además del contacto con la torre de control, el piloto dispone de radares meteorológicos a bordo, permitiéndole detectar todas estas zonas de riesgo durante el vuelo y solicitar a los controladores rutas alternativas que le permitan eludir estas situaciones adversas.

Finalmente, el último tipo de factor meteorológico importante que afecta durante el vuelo son las nubes bajas. Este tipo de nubes se forman a una baja altitud (entre los 500 y los 2000 metros) caracterizándose principalmente por su gran extensión en horizontal y su baja expansión vertical. El principal problema que presentan estas nubes es que al formarse a tan baja altura, conlleva una pérdida de visibilidad, dificultando el aterrizaje de la aeronave.

4.8.1. Meteorología en el aeropuerto

En la sección anterior se han descrito algunos de los factores meteorológicos que afectan a los aviones a lo largo de un vuelo. No obstante, estos factores también afectan al estado de las pistas de aterrizaje haciendo que estas necesiten un mantenimiento para poder ser utilizadas (ver Figura 4.9).

Es por ello por lo que la JAA (Autoridades Conjuntas de la Aviación, *Joint Aviation Authorities*, en inglés)[7] define los siguientes estados de una pista [18]:

1) **Pista seca:** Hace referencia al estado de la pista en condiciones normales, es decir, cuando no se encuentra ni mojada ni contaminada, permitiendo la realización de las operaciones con la máxima efectividad que la pista lo permite (frenado, agarre, seguridad, etc).

2) **Pista húmeda:** Estado de la pista en la que esta no se encuentra seca pero la humedad que hay en ella no afecta a su efectividad a la hora de su utilización. Se caracterizan por no tener un aspecto brillante a pesar de que existe humedad en ella.

3) **Pista mojada:** Hace referencia al estado de la pista en el que su superficie se encuentra brillante y está cubierta de agua o similar con un espesor menor a 3 milímetros. A diferencia

[7] https://jaato.com/.

de la pista seca y la pista húmeda, en la pista mojada se pierde efectividad ya que el agarre de las ruedas a la superficie de la misma es peor que en condiciones favorables.

4) **Pista contaminada:** Este estado se da cuando en más del 25 % de la superficie de la pista que se utiliza para la realización de maniobras se encuentra cubierto por agua estancada (con más de 3 milímetros de espesor), nieve o hielo. Este estado es el más inseguro de todos a la hora de operar, siendo este estado el factor determinante en el 18 % de todos los accidentes producidos durante el aterrizaje [18].

Pista seca, Fuente: *Boldmethod (visitado 16-03-2020).*

Pista húmeda, Fuente: *ShutterStock(visitado 16-03-2020).*

Pista mojada, Fuente: *https://www.slideshare.net/chococrispis37/10-contaminated-runways (visitado 16-03-2020).*

Pista contaminada, Fuente: *Pinterest (visitado 16-03-2020).*

Figura 4.9: Estados pista aterrizaje.

4.9. Aeropuerto Adolfo Suárez Madrid-Barajas

El aeropuerto "Adolfo Suárez Madrid-Barajas", también conocido como "Aeropuerto de Barajas" (IATA: **MAD**), es el aeropuerto más grande de España[8] (61.734.037 pasajeros durante el año 2019) y el 5º de Europa (por debajo de los principales aeropuertos de los países de Reino

[8] https://www.airport-technology.com/features/europes-biggest-airports/.

Unido, Francia, Holanda y Alemania). Se encuentra situado a 12 kilómetros del centro de la ciudad de Madrid y está constituido por 4 pistas dispuestas en paralelo 2 a 2 (ver Figura 4.10), sobre las que operan 82 líneas aéreas de todo el mundo[9]. Del mismo modo, dependiendo de la orientación que tengan, estas pistas se encuentran identificadas con los valores: 14L, 14R, 18L, 18R, 32L, 32R, 36L y 36R.

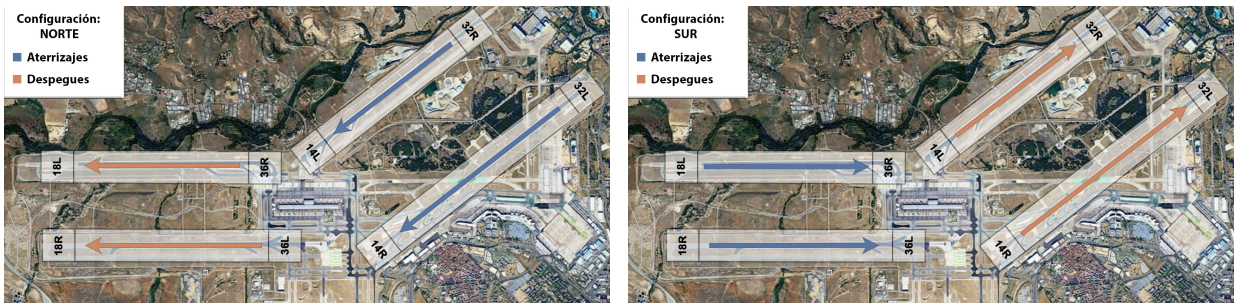
En cuanto a la configuración de las pistas, el Aeropuerto de Barajas dispone de dos configuraciones:

1) **Configuración Norte:** La configuración norte es la configuración establecida como preferente, salvo cuando por condiciones meteorológicas o por el estado de las pistas, no sea segura su utilización. La asignación de pistas para esta configuración es:

   - **32L y 32R:** Aterrizajes.
   - **36L y 36R:** Despegues.

2) **Configuración Sur:** Esta configuración se utiliza de forma auxiliar con el objetivo de no perder capacidad operativa en el caso de que ocurra alguna incidencia con la configuración Norte, por lo que se podría considerar como configuración secundaria. La asignación de pistas para esta configuración es:

   - **18L y 18R:** Aterrizajes.
   - **14L y 14R:** Despegues.



Configuración Norte Configuración Sur

Figura 4.10: Distribución de pistas y configuraciones del Aeropuerto de Barajas.

[9] https://www.aeropuertomadrid-barajas.com/lineas-aereas.php.

Capítulo 5

Aprendizaje Automático

5.1. Estado del Arte

En la última década, el número de operaciones aéreas ha aumentado exponencialmente y se prevé que esta cifra siga creciendo durante los próximos años[1]. Es por ello por lo que los investigadores se han enfocado en desarrollar herramientas, modelos y algoritmos centrados en mejorar la gestión de ese gran volumen de operaciones.

En cuanto a la configuración de las pistas se refiere, existen diferentes aproximaciones enfocadas a establecer un modelo que permita determinar qué número de pistas se utilizarán en un momento determinado, la configuración de las mismas y su utilización (aterrizaje o despegue). De esta forma se consigue proporcionar al controlador aéreo un sistema que actúe como soporte, ayudándole a la hora de determinar la configuración más adecuada. Dicha configuración será aquella que permita maximizar la capacidad operativa del aeropuerto y minimizar tanto el número de cambios en la configuración de las pistas, como el riesgo asociado a los diversos factores que influyen a la hora de su utilización.

El desarrollo de nuevas herramientas que permitan cubrir esta necesidad de una forma automática se remonta al año 2011 [19], con la creación de modelos probabilísticos que permitían obtener la configuración de las pistas del aeropuerto en base a factores logísticos, sociales y meteorológicos, siendo estos últimos los más influyentes y los que más incertidumbre introducen al modelo, debido a su gran variabilidad. No obstante, este tipo de modelos, basados únicamente en factores estadísticos, no obtienen unos resultados muy certeros (si los comparamos con modelos más recientes) y su precisión en cuanto a la elección de la configuración de las pistas ronda entre un 60-70 %. Otros enfoques similares, basados en modelos estadísticos discretos, consiguen mejorar la clasificación añadiendo una mayor complejidad al modelo. Existen aproximaciones como [4] o [5] en la que el modelo final se constituye combinando modelos de Regresión Logística Multinomial (obteniendo así la probabilidad asociada a cada configuración de pista), con modelos logísticos anidados (agrupando aquellas configuraciones que se encuentran correlacionadas), consiguiendo así una mayor precisión a la hora de pronosticar la configuración de pistas más adecuada en un determinado momento. Este tipo de modelos "combinados" obtienen una mayor precisión a la hora de determinar la configuración de las pistas, siendo esta de un 78-82 % frente al 60-70 % de un modelo "simple".

Paralelamente a esto, el avance en el campo de la inteligencia artificial propició la utiliza-

[1] https://data.worldbank.org/indicator/IS.AIR.PSGR.

ción de algoritmos y modelos de aprendizaje automático para intentar mejorar la precisión de los planteamientos basados en modelos estadísticos. Una propuesta de esto es [9] en la que se compara la precisión que se obtiene al utilizar, por un lado, algoritmos de aprendizaje supervisado para tareas de clasificación (K-NN , *K-Vecinos Más Cercanos*) y por otro lado, modelos basados en Redes Neuronales, consiguiendo una tasa de acierto de entre un 93 y un 97 %. Otra propuesta similar se da en [8] en la que, en lugar de utilizar algoritmos destinados a tareas de clasificación como el K-Vecinos Más Cercanos o los Árboles de Decisión, se emplean algoritmos de predicción, concretamente se utiliza un algoritmo de Regresión Logística Múltiple, obteniendo las distintas probabilidades que tiene cada una de las pistas de ser utilizada bajo unas determinadas condiciones (número de pistas en uso y meteorología). En general, en todas estas propuestas se tienen en cuenta aquellos factores meteorológicos que pueden llegar a condicionar la elección de una configuración u otra (dirección del viento, velocidad, visibilidad, etc), además de otros factores sociales de infraestructura (ruido, demanda, ocupación...), enriqueciendo así el modelo resultante.

No obstante, para entrenar y probar la eficacia de todos estos modelos es necesario disponer de datos históricos fidedignos y significativos, tanto de la meteorología (instante temporal, velocidad del viento, dirección y visibilidad), del aeropuerto (número de pistas, orientación, demanda, preferencias, factores sociales, utilización y configuración de pistas etcétera) y de las aeronaves (altitud, origen, destino, timestamp,...). El problema reside en que, si bien los datos de las aeronaves son públicos (se pueden obtener capturando los mensajes ADS-B), los datos de los aeropuertos o bien no lo son, o son inaccesibles (el aeropuerto de Ámsterdam-Schiphol es uno de los pocos que publican de forma abierta la configuración de las pistas[2]), dificultando así la investigación en el tema.

Es por ello por lo que se han desarrollado algoritmos que permiten suplir esa carencia de datos [21], de tal forma que, a partir de los datos ADS-B procedentes de las aeronaves, en combinación con las coordenadas geográficas de las pistas (longitud y latitud), se consigue reconstruir la trayectoria del vuelo, especialmente en los momentos de proximidad entre la aeronave y la pista (momentos de las maniobras de aproximación y de despegue). De esta forma se consigue determinar qué configuración de pista se ha utilizado para una determinada maniobra, permitiendo así generar un histórico de datos con el uso y la configuración de las pistas de un aeropuerto. A pesar de que este tipo de algoritmos no contempla la meteorología, obtiene grandes resultados en su pronóstico, consiguiendo aproximadamente un 97 % de precisión a la hora de determinar qué pista ha sido utilizada por una aeronave y qué configuración está utilizando el aeropuerto.

Finalmente, se retomará la línea de [9] empleando distintos algoritmos de aprendizaje automático supervisado que permitan abordar este problema y estudiar los resultados obtenidos para cada uno de ellos.

5.2. Aprendizaje Automático

El aprendizaje automático (*Machine Learning*, en ingles), es una de las ramas de la Informática que se centra en el diseño y desarrollo de algoritmos que permiten a los computadores mejorar su desempeño en la realización de una tarea a partir de la experiencia. La programación de estos algoritmos dista de la programación "convencional", en que aquí es el ordenador el encargado de generar estos algoritmos, ayudándose tanto de los datos de entrada como de la

[2] https://en.lvnl.nl/environment/runway-use.

salida esperada. Dicho de otro modo, el ordenador es el encargado de generar el algoritmo que permite resolver el problema, en lugar de ser una persona la que lo haga. Para ello, se suele partir de un algoritmo base, que el computador adapta al problema específico modificando diferentes parámetros que regulan su comportamiento a medida que procesa los datos de los que aprende. A estos parámetros se les conoce como **hiperparámetros** (parámetros que utiliza el modelo de aprendizaje automático para controlar el proceso de aprendizaje).

Este tipo de algoritmos son de gran utilidad a la hora de resolver problemas en los que no podemos aplicar nuestra experiencia para poder resolverlo (reconocimiento del habla), cuando los modelos se obtienen a partir de enormes cantidades de datos (análisis del genoma humano) o cuando no existe una experiencia humana previa (viajes por la Luna). Un factor muy importante a la hora de mejorar el resultado obtenido por el modelo, es la elección de las *features* (características) de los datos utilizados para entrenar. Una buena elección de las características permite que el modelo aprenda de una forma más rápida y directa, además de mejorar su eficacia. Esto se debe a que no todas las características en los datos son necesarias para llegar al resultado final y por tanto o no aportan nada de información o solo aportan confusión a la resolución del problema. Por otro lado, cabe destacar que cada uno de estos algoritmos resuelve un tipo de tarea específica, por lo que es necesario conocer el tipo de tarea antes de escoger un algoritmo para solventarla.

Desde el punto de vista del aprendizaje, existen tres tipos de aprendizaje automático:

- **Aprendizaje supervisado:** El objetivo principal de este tipo de aprendizaje consiste en enseñar o *entrenar* a un modelo a partir de datos de entrenamiento etiquetados, consiguiendo así realizar predicciones sobre datos futuros o no vistos. La principal característica es que para todas y cada una de las muestras que se utilizan en este aprendizaje se conoce la salida esperada y la estructura que tienen los datos es conocida. Las principales tareas de aprendizaje supervisado son la clasificación y la regresión.

- **Aprendizaje no supervisado:** A diferencia que en el aprendizaje supervisado, este tipo de aprendizaje trabaja con datos sin etiquetar, es decir, no se conoce la salida esperada. Además de esto, la propia estructura de los datos es desconocida, por lo que el principal objetivo de este aprendizaje consiste en explorar esos datos para obtener su estructura y obtener información significativa de ellos. La principales tareas del aprendizaje no supervisado son el clustering y la reducción de la dimensionalidad.

- **Aprendizaje por refuerzo:** El objetivo de este tipo de aprendizaje consiste en desarrollar un sistema (agente) que mejore continuamente su rendimiento a partir de interacciones con el entorno. Para ello, el entorno proporciona al agente una "recompensa" positiva o negativa en función de la acción que haya realizado permitiéndole adaptar su comportamiento y aprender las acciones que maximizan dicha recompensa.

Por otra parte, las principales tareas que van a permitir solucionar estos algoritmos son:

- **Clasificación:** Este tipo de tareas consisten en determinar a que categoría pertenece una nueva instancia desconocida, a partir de los patrones extraídos de otras instancias de la misma naturaleza aprendidas con anterioridad por el modelo. Por ejemplo, clasificar los productos de una fábrica en correctos e incorrectos.

- **Regresión:** Consiste en estimar o predecir comportamientos futuros a partir de instancias. Por ejemplo, predecir el tráfico en una ciudad, predecir el precio de los alquileres de pisos, etc.
- **Agrupamiento o Clustering:** Formar grupos a partir de instancias. Por ejemplo, agrupar páginas web en función de su contenido.

Debido a que los datos que se utilizarán están estructurados y la salida esperada es conocida, el tipo de aprendizaje empleado será el **Aprendizaje Supervisado**.

5.3. Aprendizaje Supervisado

Como se ha mencionado anteriormente, los algoritmos de aprendizaje automático supervisado se caracterizan por entrenar un modelo con datos etiquetados de tal forma que este sea capaz de clasificar o predecir ejemplos nuevos que nunca "haya visto". Es por ello por lo que es muy importante disponer de datos de ejemplo significativos, es decir, que sean un fiel reflejo de la realidad que representan (son representativos, y por tanto los patrones extraídos serán válidos para los ejemplos desconocidos; es decir, serán capaces de *generalizar* el problema), además de la salida esperada de los mismos. Una vez se dispone de este conjunto de datos se generan tres conjuntos de datos a partir de este. Esos conjuntos son:

- **Conjunto de entrenamiento (Training Set):** Este conjunto de datos se encuentra constituido por aquellos datos que se utilizarán para el aprendizaje del modelo, es decir, para el ajuste de los parámetros del algoritmo que se esté utilizando.
- **Conjunto de prueba o de test (Test Set):** Este conjunto de datos es independiente del conjunto de entrenamiento y su función es comprobar cuan bien generaliza el modelo, ya que es posible que este se encuentre sobreajustado; es decir, el modelo funciona muy bien para los ejemplos de entrenamiento pero muy mal para los nuevos ejemplos.
- **Conjunto de validación (Validation Set):** Al igual que el conjunto de entrenamiento, los ejemplos del conjunto de validación se utilizan durante el aprendizaje del modelo, concretamente para el ajuste de sus hiperparámetros. Además de esto, el conjunto de validación sirve para detectar y reducir el sobreajuste (diferencia entre la precisión obtenida en el conjunto de entrenamiento y la del conjunto de prueba) que pudiera existir. De este modo, se entrena el modelo con los datos de entrenamiento y se *valida* su funcionamiento con este conjunto de datos, con el fin de determinar los valores óptimos de sus hiperparámetros.

La distribución de los datos en estos conjuntos es sumamente importante para la eficacia del modelo final, por lo que es primordial que los datos de todos ellos sean representativos del conjunto de datos inicial, consiguiendo así una mejor eficacia y una mejor generalización por parte del modelo. Para ello existen distintas técnicas como la división por porcentajes (*Holdout*) y la validación cruzada (*K-Fold Cross-Validation*).

En el caso de algoritmos de clasificación, una vez se haya creado y entrenado el modelo hay que evaluar su rendimiento, aplicando el conjunto de prueba sobre el modelo ya entrenado. Un modelo de clasificación clasifica un conjunto de ejemplos de acuerdo a la Tabla 5.1, también llamada *matriz de confusión*. A partir de los valores de esta tabla se obtienen las métricas que se utilizarán para determinar si la eficacia del modelo es buena o no.

	Clasificado Positivo	**Clasificado Negativo**
Ejemplo Positivo	Verdadero Positivo (VP)	Falso Negativo (FN)
Ejemplo Negativo	Falso Positivo (FP)	Verdadero Negativo (VN)

Tabla 5.1: Ejemplo Matriz de Confusión.

Las métricas más habituales para la evaluación de un modelo son:

1) **Accuracy:** Representa la proporción de las muestras que han sido clasificadas correctamente de todo el total de ejemplos. El accuracy viene dado de acuerdo a la siguiente fórmula:

$$accuracy = \frac{VP + VN}{VP + VN + FP + FN} = \frac{clasificados\ correctamente}{todos\ los\ ejemplos}$$

Donde lo ideal es que *accuracy = 1*

2) **Precision:** Representa la proporción de muestras clasificadas correctamente como positivas, de todo el total de ejemplos clasificados como positivos. La precisión viene dada de acuerdo a la siguiente fórmula:

$$precision = \frac{VP}{VP + FP} = \frac{clasificados\ correctamente\ como\ positivos}{todos\ los\ clasificados\ como\ positivos}$$

Donde lo ideal es que *precision = 1*

3) **Recall:** Representa la proporción de muestras clasificadas correctamente como positivas, de todo el total de ejemplos positivos. El *recall* viene dado de acuerdo a la siguiente fórmula:

$$recall = \frac{VP}{VP + FN} = \frac{clasificados\ correctamente\ como\ positivos}{todos\ los\ positivos}$$

Donde lo ideal es que *recall = 1*

4) **F1:** Representa la media de precisión que tiene una determinada prueba o test. Esta medida de precisión viene dada de acuerdo a la siguiente fórmula:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Donde lo ideal es que *F1 = 1*

Una vez han sido definidas la divisiones del conjunto de datos y las métricas que se emplearán para determinar la eficacia de los algoritmos, se explicará más en detalle los algoritmos empleados en la Sección 5.1.

5.3.1. K-Vecinos Más Cercanos

El algoritmo K-Vecinos Más Cercanos (KNN, *K-Nearest Neighbors* en inglés) forma parte de los algoritmos de aprendizaje supervisado destinado tanto a tareas de clasificación como de regresión, siendo uno de los algoritmos más simples en sí mismo. La suposición en la que se basa es sencilla: "ejemplos similares pertenecen a la misma clase", por lo que el algoritmo asigna a un nuevo ejemplo la misma clase que tenga el ejemplo o ejemplos (depende del valor de k) más próximos. Por lo tanto, el funcionamiento de este algoritmo puede ser definido en 3 sencillos pasos:

1) Se elige un número de vecinos (k) y una medida para la distancia (distancia Euclídea, distancia de Manhattan, Similitud Coseno, etc).

2) Calcula la distancia entre los k-vecinos más cercanos de la nueva muestra a clasificar.

3) Clasifica la nueva muestra según la clase de los vecinos. Esta elección se hace mediante votación tal y como se muestra en la Figura 5.1).

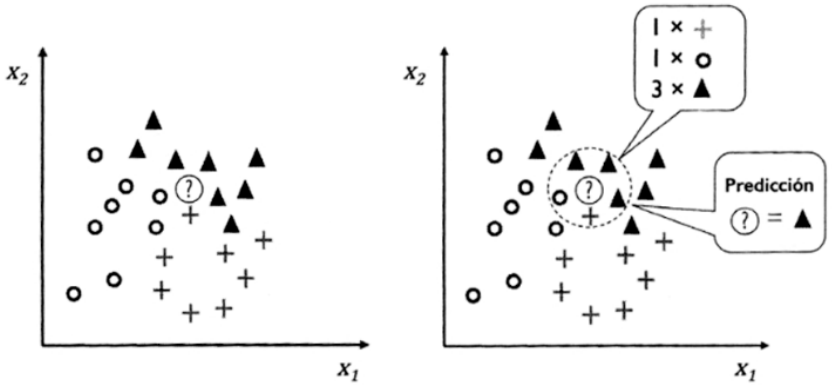


Figura 5.1: Ejemplo clasificación nueva instancia para $k = 5$. Fuente: [24].

No obstante, la elección del valor del parámetro k junto con la elección de la distancia, son cruciales para maximizar la eficacia del algoritmo: por un lado, una mala elección de k puede conllevar un sobreajuste o un subajuste del algoritmo, haciendo que este no generalice correctamente para los nuevos ejemplos. Por otro lado, una mala elección en la distancia puede hacer que esta no se ajuste correctamente a las características del conjunto de datos, perdiendo eficacia a la hora de clasificar nuevos ejemplos.

Otra característica a destacar del algoritmo de K-Vecinos Más Cercanos, es que pertenece a la categoría de modelos no paramétricos. Los modelos paramétricos se caracterizan por estimar, a partir de los datos del conjunto de entrenamiento, los parámetros necesarios para aprender una función que sea capaz de clasificar nuevos ejemplos sin la necesidad del conjunto de datos de entrenamiento original. A diferencia de este tipo de modelos, los modelos no paramétricos "memorizan" los datos del conjunto de entrenamiento y los utiliza para calcular la categoría de cada nuevo ejemplo que le llega, por lo que necesitan en todo momento disponer de todos los datos de entrenamiento para que el algoritmo funcione. Es por ello por lo que este algoritmo es un tipo de aprendizaje basado en instancias (también conocido como aprendizaje diferido).

Algunas de las ventajas del algoritmo KNN son:

- El algoritmo se **adapta** inmediatamente cuando se añaden o eliminan nuevos datos al conjunto de entrenamiento.
- Su funcionamiento es relativamente bueno en la práctica.
- No es necesario construir ningún modelo ni ajustar parámetros. Esto se debe a que se mantiene el dataset de muestras completo y únicamente se calcula la distancia que exista entre una nueva muestra y las almacenadas.

Por contra, las principales desventajas de este algoritmo son:

- Es un algoritmo lento, ya que para cada nuevo ejemplo tiene que calcular todas las distancias a cada una de las muestras que dispone. En el peor de los casos, la complejidad computacional para la clasificación de nuevas muestras crece linealmente con el número de muestras en el conjunto de entrenamiento.
- Requiere de una gran cantidad de espacio para poder almacenar el dataset de muestras completo.
- Es muy susceptible al sobreajuste debido a la maldición de la dimensionalidad[3], ya que, la variabilidad de la distancia disminuye exponencialmente con el número de dimensiones, siendo estas dimensiones el número de variables o características (*features*) de los datos que se utilicen.

5.3.2. Regresión Logística

Los modelos de regresión logística (simple y múltiple) forman parte de los algoritmos de aprendizaje supervisado destinados a tareas de regresión, aunque debido a sus características, se utiliza principalmente para tareas de clasificación, en las que las muestras u observaciones se clasifican en una categoría u otra en función de la probabilidad que tengan de pertenecer a dicha categoría.

Cabe destacar que, a pesar de que este tipo de regresión se utilice para labores de clasificación, es en esencia un modelo de regresión lineal cuya salida se modela de acuerdo al logaritmo de la probabilidad de cada grupo. Es decir, a la salida (variable de destino) se le aplica una función sigmoide para representar la probabilidad de que dicha variable pertenezca a una clase u otra, siendo clasificada en aquella cuya probabilidad sea mayor. De este modo se puede resumir la regresión logística como "regresión lineal + función sigmoide". [26]

Los modelos de regresión lineal tienen como objetivo modelar la relación entre una característica (regresión lineal simple) o múltiples características (regresión lineal múltiple), de tal forma que para a partir de una variable de entrada (variable explicativa x) se obtenga su correspondiente salida (variable de destino y). Un modelo de regresión lineal simple viene dado por la siguiente ecuación:

$$y = w_0 + w_1 x$$

Donde:

y: corresponde a la variable de destino (resultado de la regresión).

[3] https://iartificial.net/la-maldicion-de-la-dimension-en-machine-learning/.

x: corresponde a la variable explicativa (características).

w_0*:* punto de corte de la recta de regresión en el eje y.

w_1*:* coeficiente ponderado de la variable explicativa *(x)* o valor de la pendiente de la recta de regresión.

El objetivo principal consiste en aprender (encontrar) la combinación de pesos w_0 y w_1 de tal forma que se obtenga la recta que mejor se ajuste a los datos de entrenamiento, utilizando técnicas de cálculo del error (error cuadrático medio, error absoluto medio, R^2, etc.) junto con algoritmos de optimización (descenso por gradiente, Limited-Memory BFGS, etc.)

No obstante, en la mayoría de las ocasiones existen más de una variable explicativa, impidiendo la aplicación de la regresión simple para su resolución. Debido a esto, se ha de utilizar una regresión múltiple, cuyas características y planteamiento es similar al de la regresión simple, salvo que hay que añadir a la ecuación tanto las nuevas variables como los nuevos pesos asociados a estas. De esta forma, la generalización de la ecuación de la regresión simple a la regresión múltiple es la siguiente:

$$y = w_0x_0 + w_1x_1 + \ldots + w_nx_n = \sum_{i=0}^{n} w_ix_i$$

Como se puede ver en la Figura 5.2, gracias a la regresión simple se consigue la recta que mejor se ajusta a los ejemplos de entrenamiento, mientras que con la regresión múltiple se obtiene el hiperplano que mejor se ajusta a los ejemplos y cuya dimensión vendrá dada por el número de variables explicativas (características) a modelar.

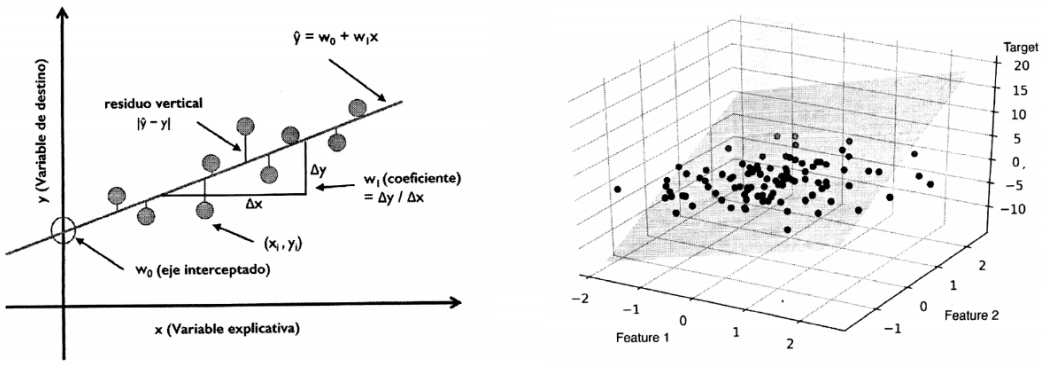


Figura 5.2: Ejemplo de recta de regresión simple e hiperplano de regresión múltiple, Fuente: [24].

Finalmente, para obtener la probabilidad asociada a cada una de las variables explicativas en función de la variable de destino, se emplea la función sigmoide, la cual viene dada por la siguiente expresión:

$$Funcion\ Sigmoide = \sigma(x) = \frac{1}{1 + e^{-x}}$$

Por ejemplo, la probabilidad de que se utilice una determinada pista (variable de destino *(Y)*) cuando la velocidad del viento (variable explicativa *(X)*) tome un determinado valor *(x)* es:

$$P(Y = k|X = x) = \frac{1}{1 + e^{-x}} = \frac{1}{1 + e^{w_0 + w_1 x}}$$

De esta forma se consigue adaptar el modelo de regresión (destinado a tareas de predicción) al modelo de regresión logística, permitiendo así realizar clasificaciones en función de la probabilidad que tenga la variable de destino (y) de pertenecer a una clase u otra, generando así un modelo de clasificación.

5.3.3. Árboles de Decisión

Los árboles de decisión son un tipo de algoritmo de aprendizaje supervisado destinado principalmente a tareas de clasificación. Este modelo se basa en una descomposición de los datos mediante una toma de decisiones que se basarán en una serie de preguntas. Los árboles de decisión están constituidos por:

1) **Nodo interno:** Los nodos internos son los atributos sobre los que se está preguntando.

   En cada nodo interno se evalúa el valor de un atributo o característica concreta, y en función del mismo se determina por qué rama se debe seguir profundizando en el árbol. Esta rama puede conducir bien a un nodo interno (donde se vuelve a evaluar el valor del atributo para determinar por qué rama seguir) o bien a un nodo hoja (arrojando el resultado final de la clasificación).

2) **Nodo hoja:** Los nodos hoja representan el valor final de la clasificación, es decir, la salida asignada al ejemplo de entrada. Habitualmente toman valores binarios (sí o no) pero también pueden ser conjuntos de valores no binarios. A estos nodos se accede a través de ramas que provienen de nodos internos y, a diferencia de estos, no tienen descendientes, por lo que son el último nodo de una rama.

3) **Rama:** Las ramas son los valores o rango de valores, que puede tomar una determinada característica residente en el nodo interno. Sirven para conectar nodos internos tanto a otros nodos internos como a nodos hoja. Un nodo interno puede tener tantas ramas de salida a otro nodo como posibles valores tenga la característica que representa.

Cabe destacar que el entrenamiento de los árboles de decisión es equivalente a la creación de los mismos, por lo que es fundamental generar el árbol de la forma más óptima posible. La clave a la hora de construir árboles de decisión reside en la forma en la que se seleccionan los atributos para hacer las divisiones del árbol ya que determinará la eficacia y la eficiencia del mismo. Una buena elección consiste en realizar la generación de los nodos internos por aquellos atributos que ofrezcan una mayor ganancia de información. De esta forma conseguimos maximizar la expresividad de la división de los datos. Además de esto, una buena práctica que se suele realizar para evitar el sobreajuste consiste en lo que se conoce como "podar el árbol". Esta acción consiste en eliminan las ramas que se encuentren por debajo de un determinando nivel de profundidad, reduciendo el nivel de detalle con que realiza la clasificación y permitiendo una mejor generalización.

Existen muchos algoritmos para la creación de árboles de decisión, los más conocidos son el ID3, C4.5, C5.0 o el CART pero el principal de todos ellos es el algoritmo ID3. Este algoritmo realiza una búsqueda por ascenso en el espacio de árboles, de tal forma que para cada nuevo

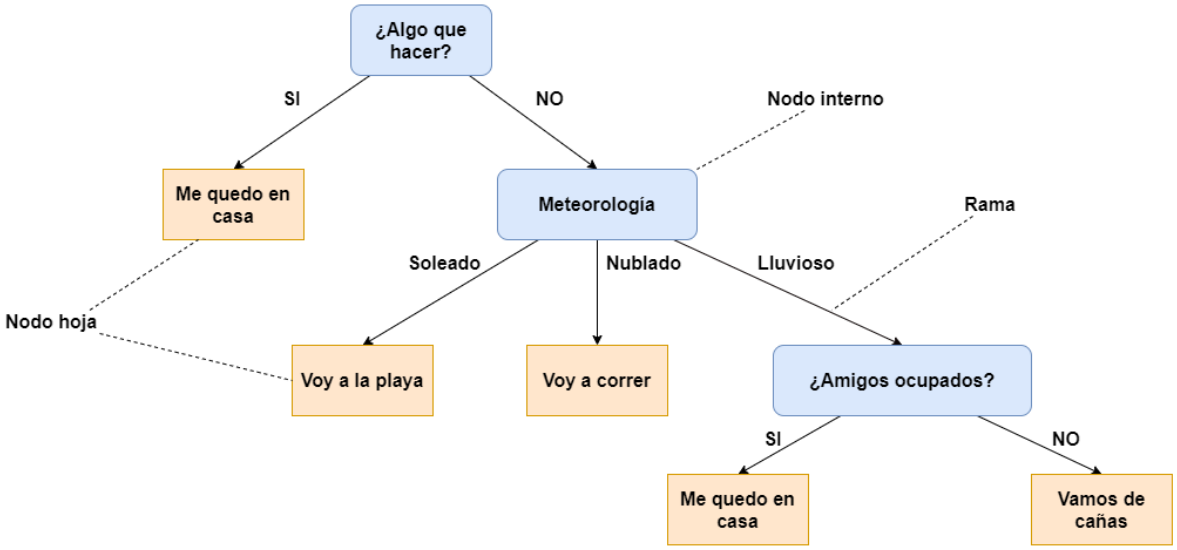


Figura 5.3: Ejemplo árbol de decisión.

nodo interno, se elige un atributo y se distribuyen los ejemplos según ese nuevo nodo interno y los anteriores. Este procedimiento se realiza hasta que se consigue que todos los ejemplos tengan el mismo valor de decisión.

La elección del atributo por el cual se va a realizar la división del nodo interno es muy importante ya que una mala elección puede hacer que se genere un árbol muy grande, produciendo una pérdida en su eficacia y su eficiencia. Es por esto por lo que el algoritmo ID3 utiliza como criterio de división que el tamaño del árbol que se forme sea mínimo. Esto se consigue haciendo divisiones sobre aquellos atributos cuya división genere nodos internos que proporcionen clasificaciones más homogéneas.

Las principales heurísticas para seleccionar los atributos por los cuales se va a realizar la división son las siguientes:

1. **Ganancia de Información:** La elección del mejor atributo de división se realiza a través de la entropía, seleccionando aquel atributo que ofrezca una mayor ganancia de información. La fórmula del cálculo de la entropía y de la ganancia de información es:

$$\text{Entropía}(S) = \sum_{i=1}^{n} -p_i \log_2 p_i$$

Donde:

S: Conjunto de ejemplos.

p_i: Probabilidad de los posibles valores.

n: Conjunto de características.

i: Valores de las características que pueden tomar los ejemplos.

$$\text{Ganancia de Información} = \text{Entropía}(S) - \sum_{v \in V(A)} \frac{|S_v|}{|S|} \cdot \text{Entropía}(S_v)$$

Donde:

S: Conjunto de ejemplos.

A: Atributos de los ejemplos.

V(A) Conjunto de posibles valores que puede tomar el atributo A.

2. **Índice de Gini:** También se conoce como impureza de Gini. Representa la probabilidad de que un elemento seleccionado aleatoriamente del conjunto de datos sea etiquetado de manera errónea, suponiendo que este ha sido etiquetado de forma aleatoria de acuerdo a la distribución de características del nuevo subconjunto. La elección del atributo se basa en aquel cuyo valor de la impureza de Gini sea mayor, ya que esto quiere decir que la homogeneidad es mayor. La impureza de Gini viene dada por la siguiente fórmula:

$$I_{Gini}(f) = \sum_{i=1}^{n} f_i(1 - f_i)$$

Donde:

f: Conjunto total de ejemplos.

f_i: Ejemplos etiquetados con la clase i del conjunto de ejemplos *f*.

En cuanto a las ventajas de los árboles de decisión destacan las siguientes:

- Fácil interpretación.
- Rapidez a la hora de clasificar nuevas muestras.
- Relativamente robustos (hasta un determinado punto no se ven afectados por los *outliers*(ejemplos poco representativos dentro del conjunto total de datos)).
- Es un modelo de caja blanca.

Por contrapartida las principales desventajas de este tipo de modelos son:

- La división por atributos que dispongan de una alta ganancia de información es crucial para que realice una buena clasificación. Si en el conjunto de datos no hay ningún atributo con dicha característica, funcionará mal.
- No es capaz de generalizar para nuevas características, por lo que si en algún momento se añade alguna característica al conjunto de datos, se ha de crear otro árbol de decisión distinto.
- Es muy fácil tender al sobreajuste para árboles de decisión demasiado profundos y específicos.

5.3.4. Ensembles

A menudo es habitual combinar diferentes modelos de clasificación en un único metaclasificador con el objetivo de obtener un mejor rendimiento de generalización que cada clasificador por separado, siendo este tipo de modelos de los más populares hoy en día para resolver problemas de aprendizaje automático. Las principales aproximaciones de *ensembles* son:

1) **Votación:** Para este tipo de *ensemble* se utilizan los mismos datos para entrenar todos los modelos que conforman el *ensemble*, y del mismo modo a la hora de realizar una clasificación, esta se pasa por cada uno de los modelos internos. Esto hace que para cada valor de entrada se obtengan n salidas intermedias (siendo n el número de modelos de clasificación que conforman el *ensemble*) y cuya salida final será aquella que haya sido predicha un mayor número de veces por estos clasificadores. Esta elección se consigue mediante el principio del **voto mayoritario** (ver Figura 5.4), cuya etiqueta de clase seleccionada es aquella que ha sido elegida por la mayoría de los modelos (en el caso de una clasificación multiclase), o la que haya recibido más del 50 % de los votos (en el caso de clasificación binaria). Existen otras técnicas para la elección de la clase final como la elección basada en porcentajes, cuya salida final viene dada por el valor máximo de la media aritmética de las probabilidades de cada etiqueta de clase por parte de cada uno de los clasificadores, es decir, suponiendo que tenemos un *ensemble* formado por 5 modelos de clasificación cuyas etiquetas de clase de salida son *casa* y *hotel*, se obtiene la probabilidad de cada una de ellas por parte de cada clasificador y se realiza la media, siendo el resultado final el mayor valor de dichas medias. Este método de voto funciona muy bien cuando los modelos que conforman el *ensemble* se encuentran bien entrenados (clasifican muy bien las muestras de forma independiente).

2) **Bagging:** Para el caso del *Bagging*, se disponen de varios algoritmos de aprendizaje automático, cada uno de ellos entrenado con un subconjunto del total de los datos de entrenamiento (*boostrap*), elegidos de forma aleatoria con repetición, de tal forma que cada uno de esos algoritmos "conoce" únicamente el subconjunto que se le ha asignado (ver Figura 5.5). Finalmente el resultado final se obtiene combinando el resultado de todos los modelos internos, pudiendo hacerse por votación, cálculo de medias aritméticas etc. El principal exponente de este tipo de *ensemble* son los **bosques aleatorios** (*random forest*, en inglés).

3) **Boosting:** En el caso del *Boosting*, los modelos que conforman este *ensemble* se entrenan de forma secuencial de tal forma que el conjunto de datos del modelo posterior contiene aquellos datos que han sido mal clasificados por el algoritmo anterior (ver Figura 5.6), de tal forma que el conjunto de datos es siempre distinto para cada uno de los modelos. La precisión de este tipo de *ensemble* suele ser mejor que la de otro tipo de *ensembles*, aunque al realizar estos pasos de forma secuencial hace que sea más lento que otras propuestas como por ejemplo el *Bagging*. Algunas aproximaciones de este tipo de *ensembles* son: AdaBoost, XgBoost o CatBoost.

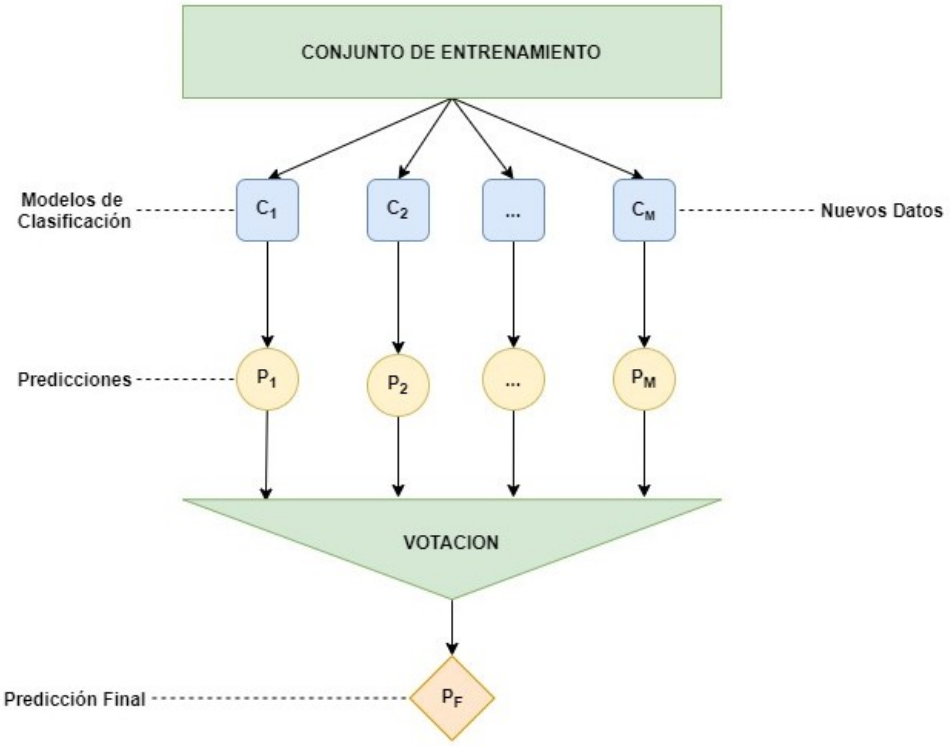


Figura 5.4: Ejemplo Votación.

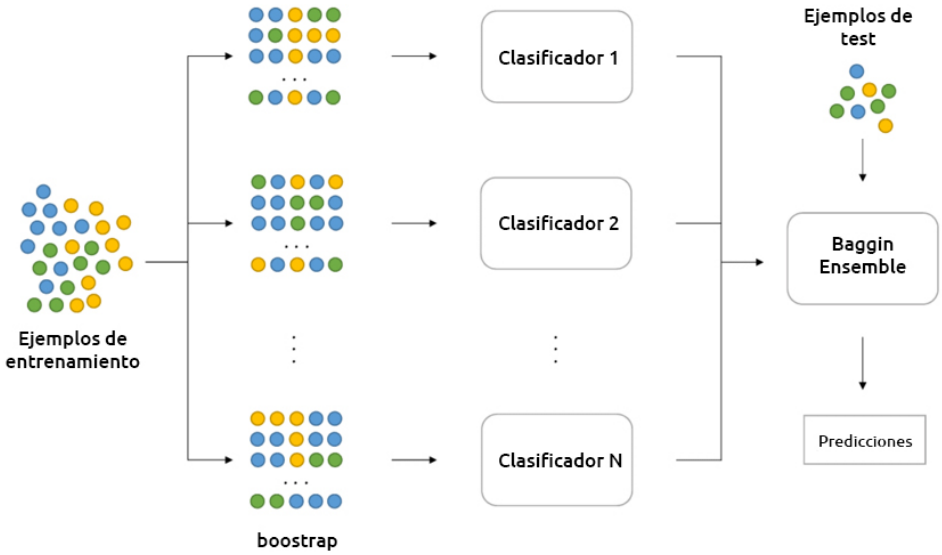


Figura 5.5: Ejemplo *Bagging*.

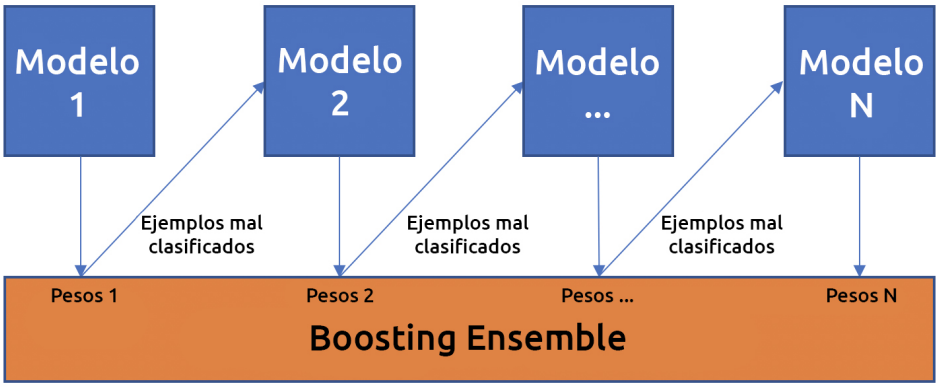


Figura 5.6: Ejemplo *Boosting*.

5.3.5. Redes Neuronales

Las redes neuronales son modelos de computación que emulan el modo en el que el cerebro humano procesa la información: gran volumen de nodos de procesamiento interconectados entre sí representando de forma abstracta una neurona. Estas unidades de procesamiento (neuronas) se organizan dentro de la red en tres tipos de capas:

1) **Capa de entrada:** Es la primera capa de la red y se encuentra constituida por todas aquellas neuronas cuya función consiste en introducir la información que se va a procesar en al interior de la red. Las neuronas que conforman esta capa no realizan ningún tipo de procesamiento.

2) **Capas ocultas:** Puede haber una o varias. Reciben como datos de entrada los datos de la salida de la capa anterior, los procesa y los envía a la siguiente capa.

3) **Capa de salida:** Es la última de las capas que conforman la red. Su función consiste en recoger la información proporcionada como salida en la última de las capas ocultas y proporcionar el resultado final del procesamiento de toda la red.

La neurona artificial se define como la unidad de procesamiento básica que trata de emular el comportamiento de una neurona humana. Además de esto, es la unidad mínima que conforma una red neuronal. Una neurona artificial está constituida por:

1) **Entrada:** Es el análogo de la dendrita en una neurona humana. Representa cada uno de los valores de entrada que va a recibir la neurona. Se representa con x_m.

2) **Peso:** Van asociados a cada una de las entradas y representa la "intensidad" o impacto que tiene la conexión de la i-ésima entrada en la neurona. Además, son los parámetros que se modifican durante el entrenamiento de la neurona para llevar a cabo el aprendizaje. Se representa con w_m.

3) **Sesgo:** El sesgo (*bias*, en ingles) es un parámetro interno dentro de la neurona que controla la predisposición de esta a devolver un valor u otro independientemente del valor de los pesos, de tal forma que, cuanto más alto sea este valor de sesgo, mayores han de ser los valores de los

pesos para poder compensarlo. Al igual que los pesos, el valor del sesgo se modifica durante el entrenamiento de la neurona. Se representa con b.

4) **Función sumatorio:** Es el núcleo de la neurona artificial. Se encarga de realizar una suma ponderada con los datos que le llegan a la neurona (entradas, pesos y sesgo), obteniendo como resultado el procesamiento de esa información. Esta función viene dada por la siguiente expresión:

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

Como se puede observar, la operación interna que realiza una neurona es la misma que la que se realiza en modelos de regresión lineal, por lo que se puede decir que internamente la neurona funciona como un modelo de regresión lineal.

5) **Función de activación:** Es la análoga a la tasa de potencial de acción de una neurona biológica. Su función consiste en proporcionar la salida de la red tomando el resultado obtenido en la suma ponderada y aplicándole una función no lineal: sigmoide, tangente hiperbólica, ReLU (*Rectified Linear Unit*), etc (ver Figura 5.7), limitando así la amplitud de la salida de la neurona. Asimismo, esta función permite distorsionar la linealidad de la salida de la neurona ya que sin esta, la salida vendría dada por el resultado de la función sumatorio, por lo que, extrapolado a una red neuronal, la salida de esta sería equivalente a la realización secuencial de operaciones de regresión lineal (tantas como neuronas tenga la red). Esta operación secuencial de regresiones lineales es equivalente a realizar únicamente una regresión lineal, por tanto, la no aplicación de esta función de activación no permitiría la creación de redes neuronales. Por lo tanto, la salida de una neurona viene dada por:

$$y = f(\sum_{i=1}^{m} x_i w_i + b) = f(z)$$

Donde:

y: corresponde a la salida de la neurona.

f(z): corresponde a la aplicación de la función de activación *f()* al resultado obtenido en la suma ponderada *(z)*.

Aprendizaje en una neurona

El aprendizaje de una neurona artificial se basa en encontrar una combinación de pesos $(w_1, w_2 \dots w_n)$ y del sesgo (b) para que el comportamiento de esta se ajuste y generalice lo mejor posible los ejemplos de entrenamiento y los de test. El procedimiento del entrenamiento de una neurona (ver Figura 5.9) es bastante simple y se puede resumir en los siguientes pasos:

1) Se inicializan los pesos y el sesgo de la neurona a 0 o a valores aleatorios pequeños: por ejemplo, valores en el rango [-0.5 , 0.5].

2) Para cada muestra del conjunto de entrenamiento $x^{(i)}$:

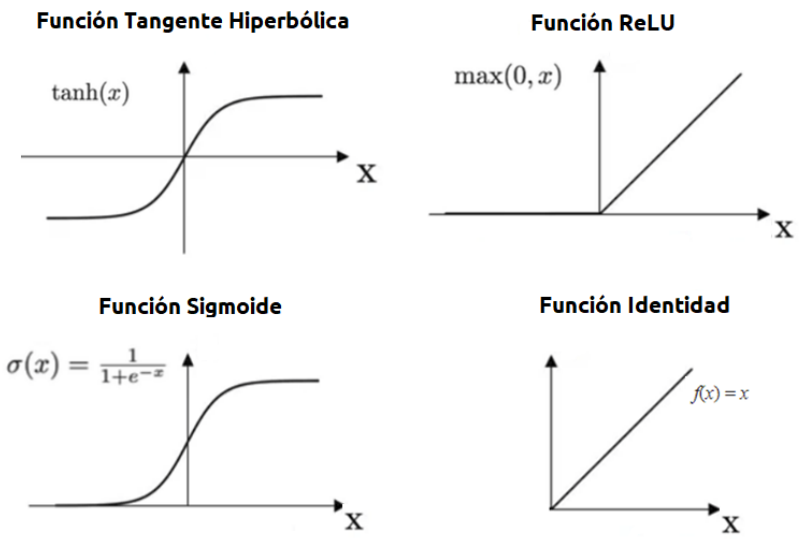


Figura 5.7: Ejemplos Funciones Activación. Fuente: *DataCamp (visitado 31-03-2020).*

a) Se clasifica la muestra con los pesos y sesgo de la red actual.

b) Se compara el resultado obtenido con el resultado esperado $(y^{(i)} - \hat{y}^{(i)})$.

c) Si hay diferencias, se calcula el ajuste (Δw_i) de los pesos y del sesgo.

$$\Delta b = \alpha(y^{(i)} - \hat{y}^{(i)})$$
$$\Delta w_i = \alpha(y^{(i)} - \hat{y}^{(i)})x_i^{(i)}$$

Donde:

Δb: Es el incremento del parámetro de sesgo.

Δw_i: Es el incremento correspondiente al peso i de la neurona.

α: Corresponde con el factor de aprendizaje que regula la velocidad de aprendizaje de la red. Es un hiperparámetro por lo que su valor rige el entrenamiento de la red. Este parámetro es una constante positiva pequeña (normalmente oscila entre 0 y 1).

$y^{(y)}$: Es la etiqueta de clase real de la muestra de entrenamiento.

$\hat{y}^{(y)}$: Corresponde con la etiqueta de clase predicha por la red.

d) Se modifican el valor de los pesos de la neurona y del sesgo.

$$b' = b + \Delta b$$
$$w_i' = w_i + \Delta w_i$$

3) Se repite el paso 2 con todos los ejemplos hasta llegar a un determinado criterio de convergencia, el cual puede venir dado por un número máximo de épocas (iteraciones sobre todos los datos del conjunto de entrenamiento) y/o un umbral para el número de ejemplos clasificados de forma errónea que se permiten.

En este punto cabe destacar que la convergencia en una neurona solo se puede garantizar siempre que ambas clases sean linealmente separables (ver Figura 5.8) y el factor de aprendizaje (α) sea suficientemente pequeño. De lo contrario, se ha de fijar alguno de los criterios de

convergencia mencionados anteriormente ya que sin ninguno de estos criterios, la red nunca pararía de actualizar sus pesos y seguiría entrenándose de manera indefinida.

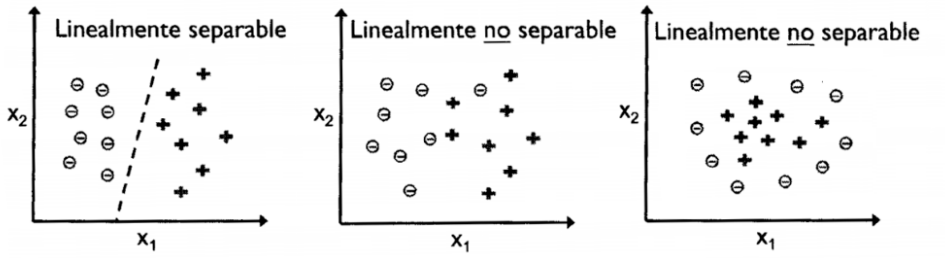


Figura 5.8: Ejemplos de conjuntos linealmente separables y no linealmente separables. Fuente: [24].

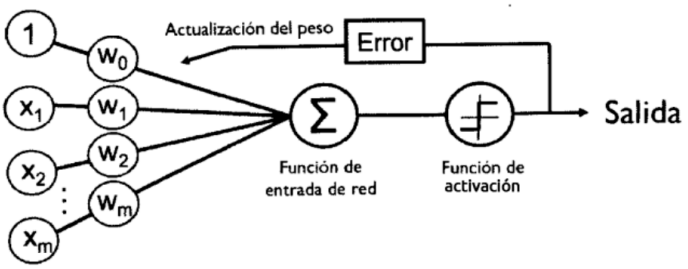


Figura 5.9: Esquema del entrenamiento de una neurona artificial. Fuente: [24].

Aprendizaje en una red neuronal

En el caso de las redes neuronales, el entrenamiento se vuelve más complicado que en una neurona ya que, como se ha mencionado anteriormente, las redes neuronales se encuentran constituidas por neuronas distribuidas en distintas capas, cuyas entradas dependen de las salidas de las neuronas que conforman la capa anterior. Por tanto, si la red devuelve una salida errónea a la esperada, ¿cómo saber qué pesos ajustar? ¿Qué neurona/s son las que se encuentran desajustadas y están propiciando que la red se comporte de forma errónea? Para resolver este problema se utilizan tres algoritmos: el algoritmo de propagación hacia adelante (*forward propagation*, en inglés), el algoritmo de propagación hacia atrás (*backpropagation*, en inglés) y el algoritmo del descenso del gradiente (*gradient descent*, en inglés).

El primer paso del entrenamiento se lleva a cabo a través del algoritmo de propagación hacia adelante, en la que cada una de las neuronas recibe los datos de entrada procedentes de la capa anterior, los procesa y los propaga a las capas posteriores hasta llegar a la última, la cual arrojará el resultado final del procesamiento de la red. Este resultado será evaluado por una función de coste[4] (raíz cuadrada media, error absoluto medio, entropía cruzada categórica, entropía cruzada binaria, etc), obteniendo el error de la red.

[4] http://www.diegocalvo.es/funcion-de-coste-redes-neuronales/.

Una vez se ha calculado el error de la red, da comienzo el siguiente paso del entrenamiento, el algoritmo de propagación hacia atrás. El funcionamiento de este algoritmo consiste en, a partir del error en la salida de la red (diferencia entre la salida de la red y la esperada), se calcula el error en cada una de las salidas de las neuronas que influyen en el resultado final. Este cálculo se realiza para cada una de las capas que conforman la red desde la última capa a la primera, de tal forma que el resultado final es una matriz compuesta por los distintos valores de la función de coste con respecto a las variaciones de los parámetros de cada una de las neuronas de las capas que conforman la red. Este procedimiento se puede resumir en 3 pasos:

1. Se calcula el error en la última capa de la red:

$$\delta^L = \frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L}$$

   Donde:

   L: Número de capas de la red.
   C: Función de coste.
   a^L: Función de activación empleada en la capa L.
   z^L: Resultado de la suma ponderada en la capa L.

2. Se realiza la retropropagación del error a la capa anterior:

$$\delta^{L-1} = W^L \delta^L \cdot \frac{\partial a^{L-1}}{\partial z^{L-1}}$$

   Donde:

   δ^L: Error producido en la capa L.
   W^L: Matriz de pesos de las neuronas en la capa L.

3. Se calcula cada una de las derivadas de la capa utilizando el error.

$$\frac{\partial C}{\partial b^{L-1}} = \delta^{L-1} \; ; \; \frac{\partial C}{\partial W^{L-1}} = \delta^{L-1} a^{L-2}$$

   Donde:

   b^{L-1}: Matriz de sesgos de las neuronas en la capa *L-1*.
   δ^{L-1}: Error producido en la capa *L-1*.
   a^{L-2}: Función de activación empleada en la capa *L-2*.

Finalmente, una vez calculado el error que se produce en la red, se aplica el algoritmo de descenso por gradiente, cuya función reside en aplicar modificaciones en los pesos de las neuronas que conforman la red, de tal forma que se minimice el valor de la función de coste, minimizando así el error (ver Figura 5.10). Este gradiente viene dado como la derivada multivariable de la función de coste con respecto a todos los parámetros de la red. El resultado gráfico de este

algoritmo es un vector que indica la dirección y sentido en el que la función de coste aumenta a un mayor ritmo, por lo que para disminuir el error, hay que avanzar en sentido opuesto a este vector. Existen distintas variantes de este algoritmo de optimización (Descenso estocástico del gradiente, AdaBoost, Adam, etc), aunque la base principal en la que se fundamentan es la misma que la que se explica en esta sección. Matemáticamente se puede expresar el descenso por gradiente de la siguiente forma:

$$\nabla f = \begin{bmatrix} \frac{\partial C}{\partial \theta_1} \\ \vdots \\ \frac{\partial C}{\partial \theta_L} \end{bmatrix} \quad ; \quad \theta = \theta - \alpha \nabla f$$

Donde:

∇f: Vector gradiente.

θ: Matriz de parámetros de la red.

θ_l: Matriz de parámetros de la red en la capa l.

C: Función de coste.

α: Factor de aprendizaje de la red.

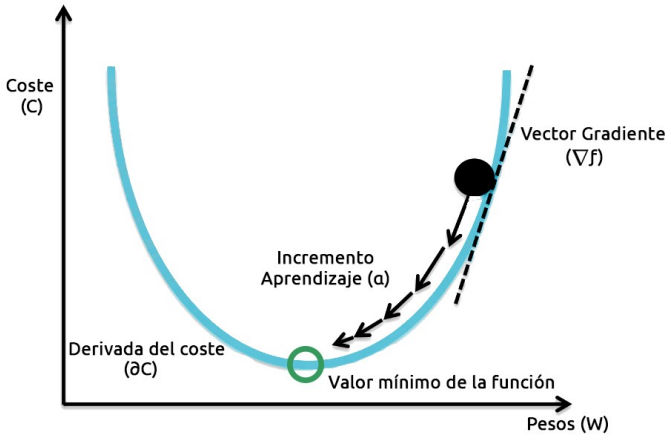


Figura 5.10: Ejemplo algoritmo descenso por gradiente.

Imaginemos que nos encontramos en una universidad formada por varios campus, cuya jerarquía se encuentra definida de la siguiente forma: 1) Rector; 2) Director de campus; 3) Director de Grado; 4) Responsable de Asignatura. Estos miembros de la jerarquía tienen como misión obtener la nota media de todas las carreras que se imparten en la universidad. Los responsables de cada asignatura se encargan de enviar al director de grado la nota media de sus asignaturas y este a su vez, toma las notas que ha recibido por parte de los responsables de las asignaturas y calcula la media de las notas en el grado, para posteriormente enviárselas al director de campus, y así sucesivamente hasta llegar al último nivel de la jerarquía, en el cual, la salida final es la

nota media de todas las carreras impartidas en la universidad. Este proceso es el equivalente al procesamiento hacia adelante en una red neuronal.

Supongamos que al final del año lectivo (época), la nota media de todas las carreras de esa universidad es un 4 (salida de la red) y el rector esperaba que fuese un 7 (salida esperada). El rector, al ver que no se han conseguido los resultados esperados, pide responsabilidades a los cargos situados un nivel inferior dentro de la jerarquía (directores de los campus de la universidad) en busca de respuestas. Estos a su vez, piden explicaciones a los que se encuentran en un nivel inferior y así sucesivamente hasta calcular la influencia-responsabilidad que ha tenido cada uno de los nodos (neuronas) dentro del sistema jerárquico (red neuronal) que han hecho que se llegue a esa situación no esperada. Esto sería el equivalente en el mundo real al algoritmo de propagación hacia atrás.

Finalmente, a partir de la información obtenida en el algoritmo de propagación hacia atrás, el jefe de control de calidad (entidad externa a la universidad), realiza un informe con aquellos cambios que se han de realizar en cada uno de los nodos que conforman la jerarquía universitaria, que van a permitir mejorar las notas y minimizar la diferencia entre lo conseguido y lo esperado (minimizar el error). Las medidas de mejora de ese informe son aplicadas a todos los nodos (neuronas) del sistema jerárquico universitario, consiguiendo así que se estos nodos se comporten de una forma distinta (ajuste de pesos) ante la misma situación (ejemplo de entrenamiento). Finalmente se vuelve a probar el mismo ejemplo, pero esta vez aplicando las medidas descritas en el informe, obteniendo así otro resultado, el cual se ajustará más a la salida deseada, minimizando así el error. Este informe y la aplicación de sus medidas a cada uno de los nodos (neuronas) del sistema jerárquico (red neuronal) es el equivalente en el mundo real del algoritmo de descenso por gradiente.

Capítulo 6

Modelo de Datos

6.1. Raw Data

Los datos son una parte fundamental para el entrenamiento y validación del modelo de aprendizaje automático, por lo que un correcto manejo de estos permitirá mejorar la eficacia final del modelo. Los datos en crudo (*raw data*) son los datos que se han obtenido en una primera instancia y sobre los que se aplicarán diversas modificaciones y adaptaciones, generando así un nuevo conjunto de datos mejorado y refinado, que será el que se utilizará en el modelo de aprendizaje final.

Como se puede ver en la Tabla 6.1, se dispone de información relativa a la operación que se realiza, el aeropuerto donde se efectúa, las condiciones meteorológicas en el momento de la operación, información relativa al vuelo, etc. Con toda este conjunto de datos en crudo se realizarán distintas operaciones de normalización y de refinamiento para poder así extraer información de ellos de una forma más eficaz, además de poder utilizarlos para entrenar modelos de aprendizaje automático ya que, en su estado en *raw*, es muy posible que o bien los modelos no funcionen correctamente o bien los resultados que den no sean tan buenos como lo serían si se hubiese realizado este pre-procesamiento inicial. Del mismo modo, se consigue asegurar que los datos son correctos: eliminando valores erróneos, nulos y *outliers*; corregir errores puntuales o sistemáticos, obtener subconjuntos de datos relevantes para el problema, etc.

6.1.1. Conjunto de datos inicial

El conjunto de datos original se encuentra constituido por registros individuales de operaciones de aterrizaje y despegue llevadas a cabo en el aeropuerto de Barajas durante los años 2018 y 2019. Estos registros se han obtenido mediante reconstrucción de trayectorias de vuelo a partir de mensajes ADS-B procedentes de las aeronaves [21].

El *dataset* original contiene un total de 733.825 registros de vuelo, de los cuales 346.198 (aproximadamente el 47,17 %) pertenecen a operaciones realizadas durante el año 2018, mientras que 387.627 (52,83 %) corresponden al año 2019. Para cada uno de estos registros se dispone de 30 características (columnas) entre las que se incluyen información relativa al momento de realizar la operación (tipo de operación, altitud, velocidad, etc), a la meteorología en el aeropuerto (velocidad del viento, lluvia, visibilidad, etc) y al vuelo (identificador de la aeronave, origen, destino, día en el que se realizó el vuelo, etc). En las Tablas 6.2, 6.3 y 6.4 se muestran las principales propiedades (tipo de datos, número de registros nulos, valor mínimo, máximo y medio,

Campo	Descripción
id_leg	Identificador de la trayectoria.
runway	Pista utilizada.
operation	Operación que se realiza: (Landing, Take-Off).
hexid	Identificador del transpondedor de la aeronave.
callsign	Identificador del vuelo.
time_stamp	Instante temporal en el que comienza la operación. (aterrizaje/despegue).
altitude	Mediana de las altitudes obtenidas en los mensajes ADS-B referidas a la operación.
ground_speed	Mediana de las velocidades obtenidas en los mensajes ADS-B referidas a la operación (respecto a tierra).
vertical_rate	Mediana de las velocidades de ascenso y descenso obtenidas en los mensajes ADS-B referidas a la operación.
type	Tipo de aeropuerto.
origin	Origen del vuelo.
destination	Destino del vuelo.

Campo	Descripción
part_date_utc	Fecha en la que se realizó el vuelo.
valid_time	Fecha y hora en la que se emitió el reporte meteorológico del METAR.
tmp	Temperatura.
dew_pt	Punto de rocío.
rel_hum	Humedad relativa.
wind_dir	Dirección del viento.
win_sp	Velocidad del viento.
wind_gust	Velocidad de las rachas de viento.
visib	Visibilidad.
press	Presión.
rain	Existencia de lluvia.
snow	Existencia de nieve.
drizzle	Existencia de llovizna.
fog	Existencia de niebla.
time_class	Hora del día en la que se realizó la operación.
time_ref	Intervalos de 30 minutos transcurridos desde las 00:00.
day_week	Día de la semana.
month	Mes.

Tabla 6.1: Descripción de las características del modelo de datos inicial.

etc) de los atributos pertenecientes a cada una de estas entidades:

Atributo	Tipo de datos	Registros nulos	Media	Desviación estándar	Valor mínimo	Valor máximo
callsign	String	2	–	–	–	–
hexid	String	1	–	–	–	–
id_leg	String	0	–	–	–	–
origin	String	2.296	–	–	–	–
destination	String	34.868	–	–	–	–
time_stamp	String	1	–	–	–	–
part_date_utc	String	1	–	–	–	–
valid_time	String	1	–	–	–	–
day_week	Float	1	2,96	± 1,97	0	6
month	Float	1	6,69	± 3,37	1	12

Tabla 6.2: Análisis características pertenecientes al vuelo.

Atributo	Tipo de datos	Registros nulos	Media	Desviación estándar	Valor mínimo	Valor máximo
operation	String	1	–	–	[Landing; Take-Off]	
altitude	Float	1	2.985,29	± 856,99	1.350	6.400
ground_speed	Float	1	158,42	± 20,65	0	730
vertical_rate	Float	1	719,41	± 1.506,66	-15.264	8.128
type	String	1.039	–	–	–	–
runway	String	1	–	–	[14L;14R;18L;18R; 32L;32R;36L;36R]	

Tabla 6.3: Análisis características pertenecientes a la operación.

Atributo	Tipo de datos	Registros nulos	Media	Desviación estándar	Valor mínimo	Valor máximo
time_class	Float	1	12,88	± 5,18	0	23
time_ref	Integer	0	18.269,42	± 9.975,01	0	35.040
tmp	Float	29.968	17,28	± 9,34	-6	41
dew_pt	Float	29.968	6,14	± 5,24	-16	20
rel_hum	Float	29.968	55,13	± 25,17	4,98	100
wind_dir	Float	110.476	199,71	± 108,33	0	360
wind_sp	Float	30.002	16,11	± 103,93	0	3240
wind_gust	Float	3.033	3,43	± 11,87	0	90,74
visib	Float	32.013	9,76	± 1,03	0	10
press	Float	29.968	1.016,84	± 6,71	982	1.036
rain	Float	3.033	-0,02	± 0,14	-1	1
snow	Float	3.033	$-6,3 \cdot 10^{-5}$	± 0,007	-1	0
drizzle	Float	3.033	$-8 \cdot 10^{-3}$	± 0,092	-1	1
fog	Float	3.033	0,01	± 0,13	0	1

Tabla 6.4: Análisis características pertenecientes a la meteorología.

6.2. Generación del conjunto de datos final

La generación del conjunto de datos final se ha llevado a cabo tras un proceso de refinado sobre los datos en crudo (*raw data*), de tal forma que, aplicando una serie de transformaciones sobre estos se consiga una mejora significativa a la hora de utilizarlos para el entrenamiento de modelos de aprendizaje automático. Las transformaciones que se han llevado a cabo sobre los datos han sido las siguientes:

1) **Normalización de las variables cualitativas:** Las variables cualitativas o variables nominales, son un tipo de variables estadísticas que permiten la representación o descripción de cualidades sin la necesidad de utilizar valores numéricos, como por ejemplo, la representación del sexo de una persona (hombre, mujer), el estado civil (soltero, casado, divorciado, etc).

   Este tipo de variables se caracterizan principalmente por no poder medirse u ordenarse numéricamente, por lo que no son interpretables según los principios en los que se basan los

algoritmos de aprendizaje automático (matemáticos y estadísticos), para lo cual es necesaria su transformación a un valor numérico. Por ello es necesario llevar a cabo un proceso de normalización-codificación de estas variables, de tal forma que se consiga representar de forma numérica esta información, haciendo posible su utilización. Para llevar a cabo este proceso de normalización se han utilizado las siguientes técnicas:

a) **Codificación mediante etiquetas de clase:** Este tipo de codificación es la más sencilla e intuitiva a la hora de realizar un mapeo de clases. El funcionamiento de este tipo de codificación consiste en asignar una etiqueta numérica (generalmente comienza en el valor 0) a cada uno de los valores únicos de una columna, de tal forma que si tuviésemos una columna ***Sexo***, cuyos valores fueran: *Masculino, Femenino y Otro*, su codificación sería: *{"Masculino": 0, "Femenino": 1, "Otro": 2}*, obteniendo así datos cuantitativos a partir de datos cualitativos.

b) **Codificación en caliente (hot encoding):** La codificación en caliente (*hot encoding*, en inglés) es un tipo de codificación de variables cualitativas cuya idea consiste en generar una característica ficticia para cada valor único en la columna de características nominales, de tal forma que el resultado final estaría formado por tantas columnas como valores únicos hubiese. Siguiendo el ejemplo anterior, la codificación en caliente de la columna ***Sexo*** (cuyos valores únicos son: *Masculino, Femenino u Otro*) estaría constituida por 3 nuevas columnas: *(Sexo_ Masculino, Sexo_ Femenino, Sexo_ Otro)*, cuyos valores serían (1,0,0) para el sexo Masculino; (0,1,0) para el Femenino y (0,0,1) para Otro. Las columnas resultantes de esto se las conoce también como *variables dummies* (*dummy variables*, en inglés).

Columnas Originales		**Etiquetas de Clase**	**Codificación en Caliente**		
Nombre	**Sexo**	**Sexo Etiqueta**	**Sexo Masculino**	**Sexo Femenino**	**Sexo Otro**
Catelyn	Femenino	1	0	1	0
Caminante Blanco	Otro	2	0	0	1
Arya	Femenino	1	0	1	0
Eddard	Masculino	0	1	0	0
Tyrion	Masculino	0	1	0	0

Tabla 6.5: Ejemplo codificación mediante etiquetas de clase y codificación en caliente.

2) **Tratamiento de datos ausentes:** Es habitual que a la hora de trabajar con datos, nos encontremos con que faltan valores. Esto puede deberse a determinadas causas (error en el proceso de recopilación de datos, que algunos valores se obtengan mediante la combinación de otros y esta no se haya podido llevar a cabo para algunos registros, etc). Esto supone un problema a la hora de trabajar con el dataset, ya que, tanto a la hora de obtener información de ellos como a la hora de operar, podemos llegar a propagar esa falta de datos a otras columnas, obtener conclusiones erróneas de los datos a la hora de analizarlos, afectar al correcto funcionamiento del modelo de aprendizaje automático y otras muchas contrapartidas debido a esta ausencia. Es por ello por lo que solventar esta falta de datos es una tarea

primordial y que se ha de realizar antes de efectuar cualquier otra operación sobre el conjunto de los datos. Las principales técnicas de tratamiento de datos ausentes y las que se han utilizado son las siguientes:

a) **Eliminación de muestras o características:** La eliminación de muestras o características, es la técnica más sencilla a la hora de operar con datos ausentes. Consiste llevar a cabo la eliminación de todos aquellos registros para los cuales no disponemos de todas sus características (para el caso de la técnica de eliminación de muestras), o bien eliminar todas aquellas columnas en las que existe algún valor ausente. A pesar de que es la técnica de tratamiento de datos ausentes más básica e intuitiva, también es la que más impacto causa ya que durante el proceso se pierde una grandísima cantidad de datos que pueden ser valiosos, además de perder características importantes. Es por ello por lo que es recomendable utilizarla únicamente en situaciones muy puntuales o específicas y optar por otro tipo de técnicas.

b) **Interpolación de datos ausentes:** A menudo no es posible realizar una eliminación de muestras o características ya que podríamos llegar a la situación de perder la mayor parte de los datos o incluso todos. La solución a este problema consiste en realizar una imputación de los datos ausentes. Las principal técnica de interpolación de datos ausentes es la **técnica de imputación de datos**. Esta técnica consiste en asignar a cada valor ausente de una columna un nuevo valor, el cual se obtiene a partir de los datos de esa columna. Los valores más utilizados para asignar a los valores desconocidos son: 1) Media de todos los valores; 2) Valor máximo, 3) Valor mínimo, 4) Primer valor conocido antes del valor ausente; 5) Primer valor conocido después del valor ausente; 6) Valor por defecto: (0, 1, etc).

6.2.1. Obtención de la configuración a partir de utilización de las pistas

Debido a que en el conjunto de datos en bruto no disponemos de ninguna característica explícita de la configuración de pistas que se está utilizando en cada uno de los vuelos, será necesario llevar a cabo un proceso de obtención de nuevas características a partir de los datos originales. Esto supone un paso adicional en el tratamiento de los datos ya que es necesario realizar operaciones sobre estos con la finalidad de obtener una nueva columna que a priori no existe. Para ello se ha realizado un proceso de extracción de información a partir de los datos iniciales y es que, como ya se ha mencionado en la Sección 4.9, el aeropuerto de Barajas dispone de dos configuraciones: Configuración Norte y Configuración Sur. Cada una de estas configuraciones tienen asignadas un determinado número de pistas para realizar las maniobras de aterrizaje y otras para hacer las maniobras de despegue. Con esta información y la disponible en el dataset, es posible determinar cuántas operaciones de cada tipo han sido realizadas en cada una de las pistas durante un intervalo determinado de tiempo, consiguiendo así determinar qué configuración se ha estado utilizando en cada instante temporal y, por tanto, obteniendo la configuración en la que cada vuelo ha realizado la maniobra correspondiente.

Para ello se ha realizado un análisis en tres fases:

1) **Análisis de la configuración a nivel mensual:** El primer análisis realizado consistió en observar el comportamiento de la configuración del aeropuerto para cada uno de los meses. De esta forma se pretende encontrar algún patrón de comportamiento para cada uno de los

mismos, ya que, hay meses concretos que coinciden con el periodo de vacaciones y por tanto supone un aumento en el número de operaciones que se realizan.

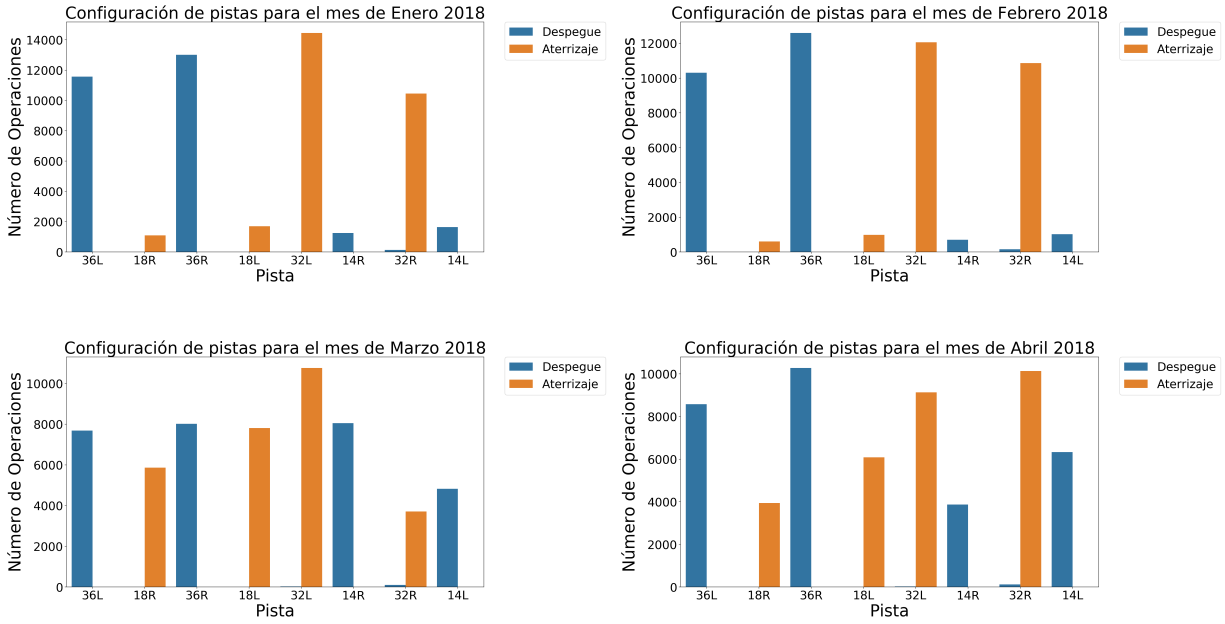


Figura 6.1: Configuración de las pistas para los meses de Enero, Febrero, Marzo y Abril (en azul: despegue; en naranja: aterrizaje).

Como se puede observar en la Figura 6.1, la idea de intentar observar la configuración a una escala temporal tan general no nos permite obtener los instantes temporales en los que se producen estos cambios de una configuración a otra ya que estos se realizan en intervalos temporales relativamente pequeños (entre 15 y 45 minutos), por lo que para obtener unos valores más representativos es necesario reducir el intervalo temporal sobre el que se realiza el análisis.

2) **Análisis de la configuración a nivel de horas y minutos:** Debido a que el intervalo sobre el que se estaba obteniendo la configuración era demasiado amplio, se disminuyó hasta periodos de 1 hora y de 30 minutos. La diferencia principal entre estos es que, como era de esperar, con el intervalo de 30 minutos se consigue una mayor precisión a la hora de localizar el momento temporal exacto en el que se realiza un cambio de una configuración a otra, por lo que, este último será el intervalo que se utilizará para determinar la configuración.

   Como se puede ver en la Figura 6.2, para este intervalo se aprecia claramente la configuración que hay en cada uno de estos instantes temporales, consiguiendo así una mayor precisión a la hora de determinar los cambios. No obstante, hay intervalos en los que se están utilizando pistas pertenecientes a ambas configuraciones. Esto se debe a que, posiblemente en ese periodo temporal se esté produciendo un cambio entre una configuración y otra ya que no puede haber dos configuraciones al mismo tiempo. Para filtrar en una primera instancia estas situaciones y eliminar el posible ruido que pudiera haber en alguna de ellas, se ha establecido un valor de sensibilidad, el cual determinará si una configuración en un intervalo conflictivo es norte o sur

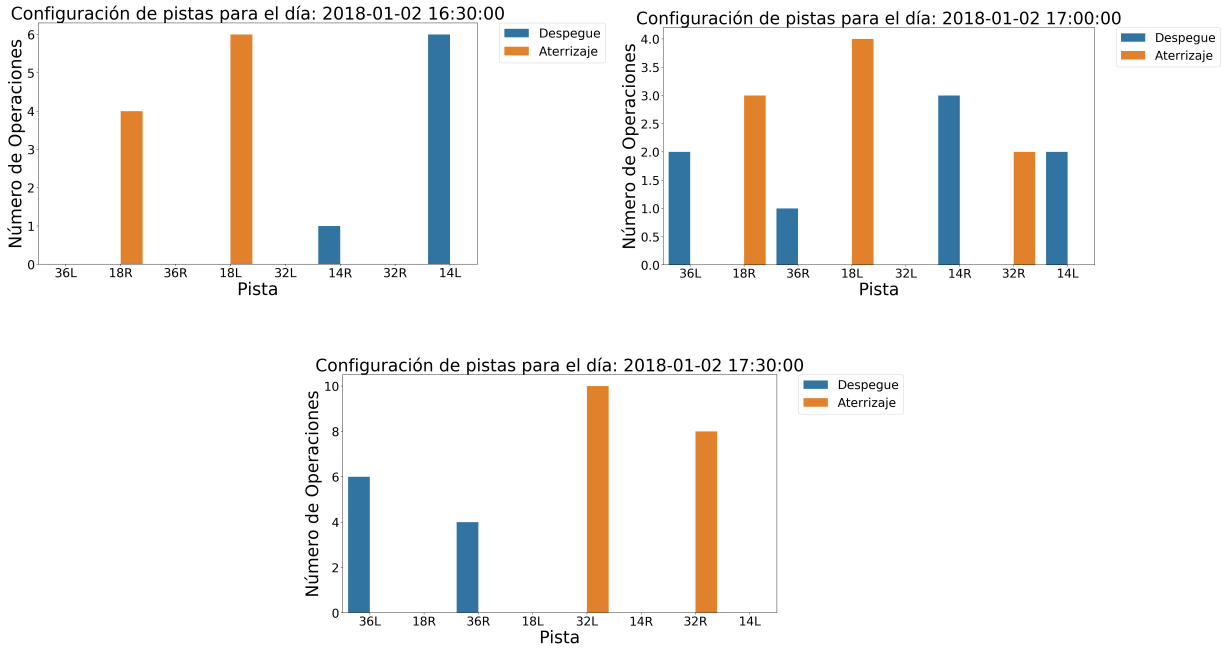


Figura 6.2: Configuración en intervalos de 30 minutos (en azul: despegue; en naranja: aterrizaje).

en función de dicho valor, por ejemplo, si tenemos un valor de sensibilidad del 60 %, un instante temporal en el que se utilicen ambas configuraciones se considerará como Configuración Norte si al menos el 60 % de los vuelos en dicho intervalo se realizan en las pistas pertenecientes a la Configuración Norte, y del mismo modo con la Configuración Sur. En el caso de que ninguna configuración consiguiese superar dicho umbral, se mantiene como "configuración indeterminada" solventándose posteriormente.

Tras llevar a cabo este análisis sobre los 36.743 intervalos de tiempo, la detección de intervalos conflictivos ha sido:

Sensibilidad	**Intervalos concflictivos**	**Vuelos Afectados**	**Vuelos afectados (%)**
50 %	22	176	0,023
60 %	125	2.295	0,312
75 %	295	5.643	0,768
95 %	513	10.198	1,389

Tabla 6.6: Impacto de los intervalos conflictivos en los vuelos en función de la sensibilidad.

La Figura 6.3, representa el número de intervalos conflictivos a medida que aumenta el valor de la sensibilidad.

Finalmente, para eliminar los intervalos conflictivos que han quedado tras el filtrado por el umbral de sensibilidad, se han tomado las siguientes medidas:

a) **Reducción del intervalo en los puntos conflictivos:** La primera medida tomada para eliminar esta característica y mejorar la precisión en cuanto al instante temporal en el que

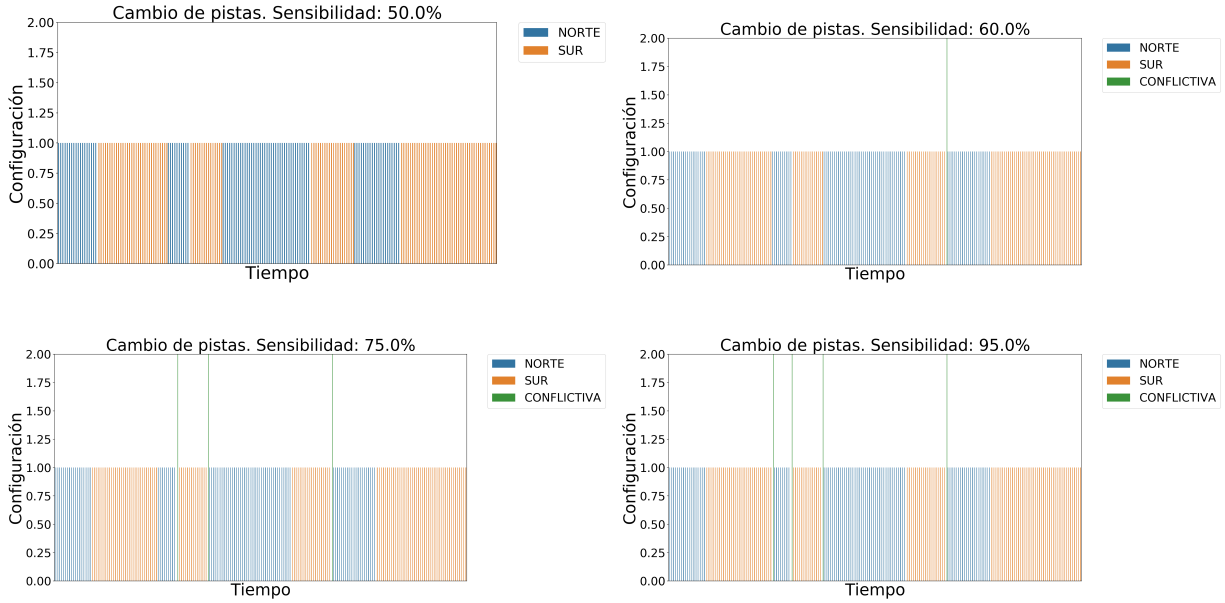


Figura 6.3: Detección de configuraciones conflictivas en función del umbral de sensibilidad.

se realizó el cambio de una configuración a otra, fue la de reducir a 15 minutos el intervalo en los puntos en los que aparecía que se estaban utilizando las dos. Esta medida no fue muy efectiva ya que en la mayoría de estos puntos no se disponía de registros de vuelos con 15 minutos de antelación o retraso. No obstante, gracias a ella se consiguió mejorar la configuración para 1.952 de los vuelos.

b) **Mayor utilización de pistas:** Esta medida consiste en comparar el número de operaciones que se realizan sobre las pistas pertenecientes a la Configuración Norte con el número de operaciones que se realizan en las pistas de la Configuración Sur, asignando a cada periodo conflictivo la configuración con una utilización de pistas mayor. A pesar de que esta modificación puede resultar intuitiva, no deja de ser una sustitución importante sobre el conjunto de los datos, por lo que, es importante observar el impacto global que esta supone. Esta medida afecta a un total de 8.246 vuelos (aproximadamente un 1,13 % del conjunto de los datos), por lo que suponiendo que esta fuera muy perjudicial, el impacto que supone es mínimo.

Finalmente, con los pasos descritos a lo largo de esta subsección, ya tenemos la información necesaria para generar la nueva columna con la configuración de la pista que se está utilizando. El siguiente y último paso consiste en tomar la hora de realización de la operación en cada uno de los registros de los vuelos y asignarle a este la configuración utilizada en ese momento, generando así una nueva columna a partir de los datos del dataset inicial. Sobre esta columna aplicaremos las técnicas de normalización de variables cualitativas, obteniendo una nueva columna que describe (de forma normalizada) la configuración de las pistas para cada uno de los registros de vuelo. Esta nueva columna normalizada será la columna a predecir por el modelo de aprendizaje automático.

6.3. Conjunto de datos final

Una vez finalizada la tarea de preprocesado de los datos en crudo, en la que se han realizado operaciones de normalización, tratamiento de valores ausentes y generación de nuevas características necesarias a partir de los datos disponibles, es el momento de unificar todos esos datos y generar así el conjunto de datos final. Este conjunto de datos será el que se utilizará a lo largo del proyecto tanto para entrenar y validar los modelos de aprendizaje automático, como para seleccionar la mejor configuración de los hiperparámetros del mismo, consiguiendo de este modo evaluar la eficacia de este.

En la Tabla 6.7 se muestran tanto las nuevas características introducidas en el modelo de datos, como aquellas que han sido eliminadas de este, de tal forma que que el modelo de datos final queda conformado y estructurado de la forma en que se muestra en la Figura 6.4.

Campo	Tipo de característica	Modificación realizada
time_stamp	Original	- Eliminación
part_date_utc	Original	- Eliminación
valid_time	Original	- Eliminación
operation_Landing	Artificial	- Hot Encoding
operation_TakeOff	Artificial	- Hot Encoding
runway_14L	Artificial	- Hot Encoding
runway_14R	Artificial	- Hot Encoding
runway_18L	Artificial	- Hot Encoding
runway_18R	Artificial	- Hot Encoding
runway_32L	Artificial	- Hot Encoding
runway_32R	Artificial	- Hot Encoding
runway_36L	Artificial	- Hot Encoding
runway_36R	Artificial	- Hot Encoding
configuracion_NORTE	Artificial	- Hot Encoding
configuracion_SUR	Artificial	- Hot Encoding
configuracion	Artificial	- Etiquetas de Clase

Tabla 6.7: Modificaciones realizadas para llevar a cabo la generación del modelo de datos final.

6.4. Extracción de información del modelo de datos

Después de haber generado el modelo de datos final, asegurándonos de que todos los datos se encuentran normalizados y que no hay ningún valor ausente, el siguiente paso consiste en realizar un análisis de los datos en busca de relaciones y dependencias entre las características de los datos que se utilizarán como *predictors* (atributos que se utilizarán para ayudar al modelo de aprendizaje automático a obtener el resultado esperado) y el *target* (atributo o característica que se quiere predecir y que actúa como salida del modelo de aprendizaje automático). Para ello la forma más eficaz y visual de hacerlo es mediante la utilización de una matriz de correlación[1].

[1] https://support.minitab.com/es-mx/minitab/18/help-and-how-to/modeling-statistics/multivariate/how-to/item-analysis/interpret-the-results/all-statistics-and-graphs/.

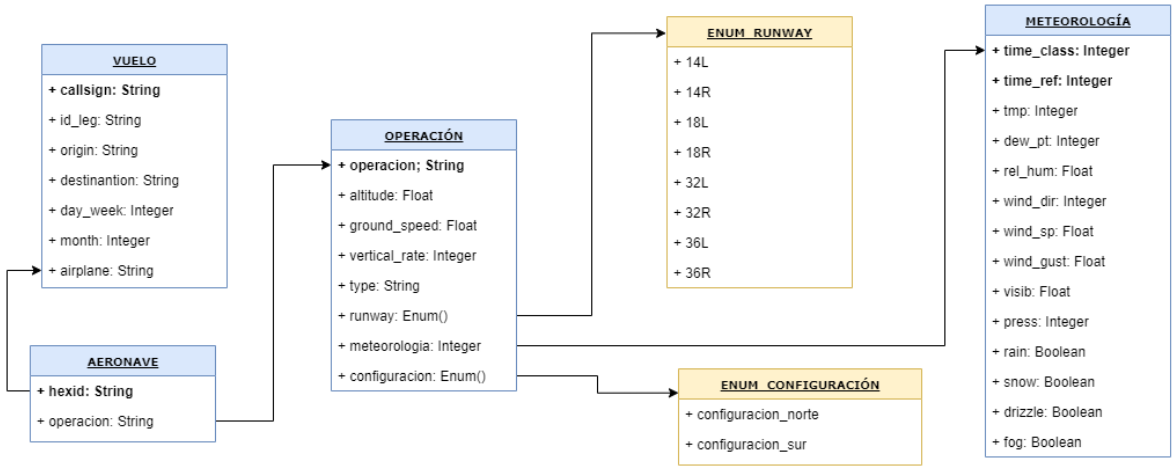


Figura 6.4: Diagrama de clases.

Una matriz de correlación es una tabla de doble entrada en cuyos ejes se sitúan las etiquetas o los nombres de los distintos atributos de los que se desea obtener la correlación, y cuyos valores se corresponden con el coeficiente de correlación (ρ) (generalmente se utiliza el coeficiente de *Pearson* pero se puede utilizar otro coeficiente como el de *Kendall* o el de *Spearman*). Este coeficiente muestra la relación o independencia que tiene una variable con respecto de otra en un rango de valores que va desde el -1 hasta el 1. Dependiendo del valor que tome, se puede interpretar de la siguiente forma:

- ($\rho = 1$): Dependencia total positiva entre ambas variables (cuando una de ellas aumenta, la otra también lo hace en una **proporción constante**).
- ($0 < \rho < 1$): En este caso tenemos una dependencia positiva entre ambas variables (cuando una de ellas aumenta, la otra también lo hace).
- ($\rho = 0$): Independencia total entre ambas variables.
- ($-1 < \rho < 0$): En este caso tenemos una dependencia negativa entre ambas variables (cuando una de ellas aumenta, la otra disminuye).
- (ρ = -1): Dependencia total negativa entre ambas variables (cuando una de ellas aumenta, la otra disminuye en una **proporción constante**).

Como se puede apreciar en la matriz de correlación de la Figura 6.5, los principales atributos que identifican tanto a la aeronave como al vuelo, no tienen relación alguna ni con la pista que utiliza ni con la configuración. Este hecho es razonable ya que la elección de las pistas se hace de acuerdo a condiciones meteorológicas o de tráfico aéreo y en ningún caso un determinado vuelo o aeronave tiene "reservada" una pista para ser utilizada únicamente por ella, por lo que es lógico que estas variables no tengan relación ni directa ni indirecta sobre la pista o la configuración. La conclusión a la que podemos llegar mediante esta matriz es que el origen, destino y aeronave que realiza un vuelo no influyen a la hora de determinar en qué pista ha de realizar este la maniobra.

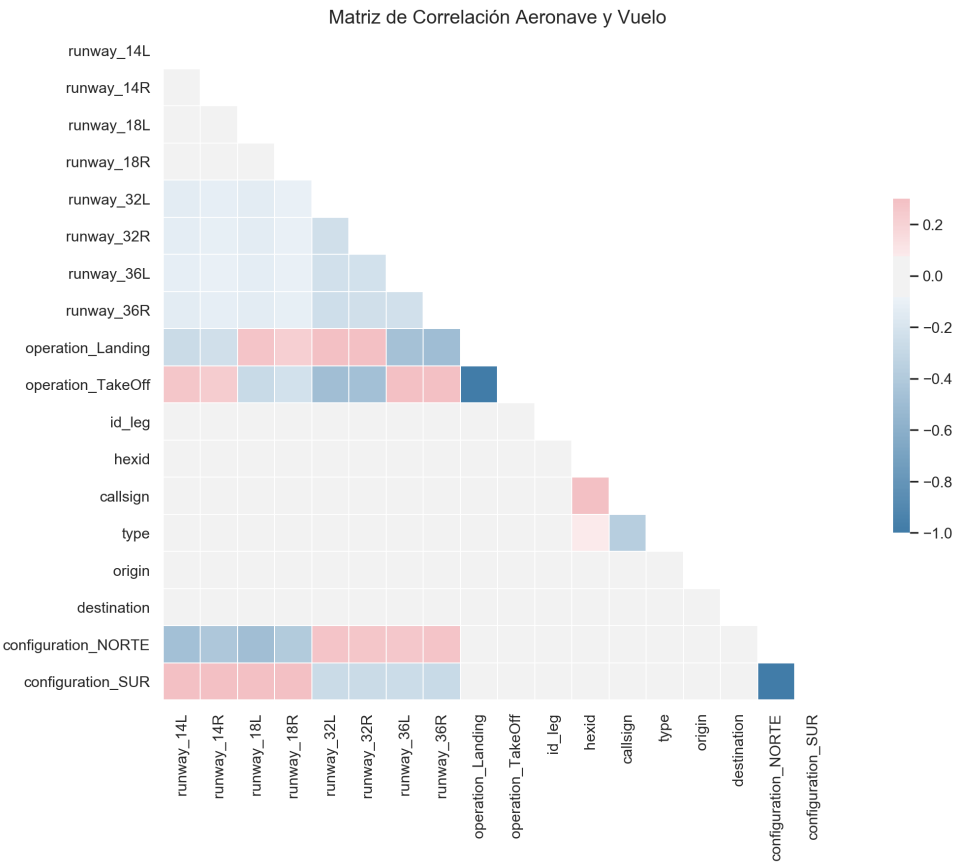


Figura 6.5: Matriz de Correlación para todos los atributos de la aeronave y el vuelo.

Si nos fijamos ahora en cómo se encuentran relacionadas las pistas y las configuraciones con respecto a los atributos relativos a la hora en la que se realiza la maniobra (ver Figura 6.6), vemos como aquí sí que existen más relaciones entre los atributos propios de la maniobra, el estado de las pistas y la configuración. En este caso se puede observar que existe una fuerte correlación positiva entre la pista 14L, la Configuración Sur, la altitud, la operación de despegue y las velocidades tanto vertical como con respecto a tierra, confirmando de este modo la utilización de dicha pista para la realización de las maniobras de despegue. De forma análoga podemos llegar a la misma conclusión para el resto de las pistas y configuraciones. Las conclusiones a las que podemos llegar mediante esta matriz son las siguientes:

a) La Configuración Norte se encuentra constituida por las pistas 32L, 32R, 36L y 36R.

b) La Configuración Sur se encuentra constituida por las pistas 18L, 18R, 14L y 14R.

c) Las pistas 32L, 32R, 18L y 18R se utilizan para realizar las maniobras de aterrizaje.

d) Las pistas 36L, 36R, 14L y 14R se utilizan para realizar las maniobras de despegue.

Finalmente, en el caso de la matriz de correlación entre la meteorología, las pistas y la configuración (ver Figura 6.7), observamos cómo existe una correlación positiva entre la pistas:

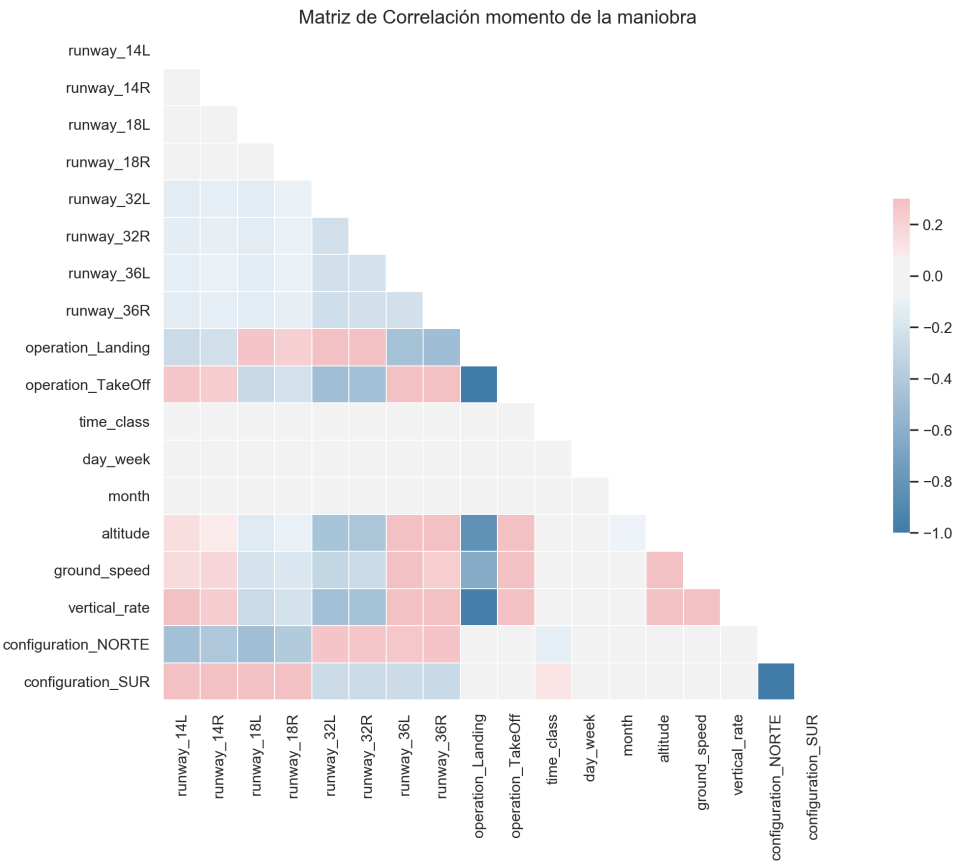


Figura 6.6: Matriz de Correlación para los atributos a la hora de realizar la maniobra.

14L, 14R, 18L, 18R, la temperatura y el viento (velocidad y rachas). Esto nos indica que en el caso de que existiesen rachas de viento en el momento de la maniobra, estas se efectuarían siempre en alguna de las pistas anteriormente mencionadas, motivado por motivos de seguridad a la hora de realizar la maniobra. En caso contrario, las maniobras se realizarían sobre las pistas restantes: 32L, 32R, 36L y 36R. Las conclusiones a las que podemos llegar mediante esta matriz son las siguientes:

a) En caso de que aumente la velocidad del viento y las rachas, las maniobras se realizarán sobre las pistas: 14L, 14R, 18L, 18R.

b) En caso de que disminuya la velocidad del viento y las rachas, las maniobras se realizarán sobre las pistas: 32L, 32R, 36L y 36R.

c) En el caso de que se produzca un aumento en la temperatura, las maniobras se realizarán sobre las pistas: 32L, 32R, 36L y 36R.

d) En el caso de que se produzca una disminución en la temperatura, las maniobras se realizarán sobre las pistas: 14L, 14R, 18L, 18R.

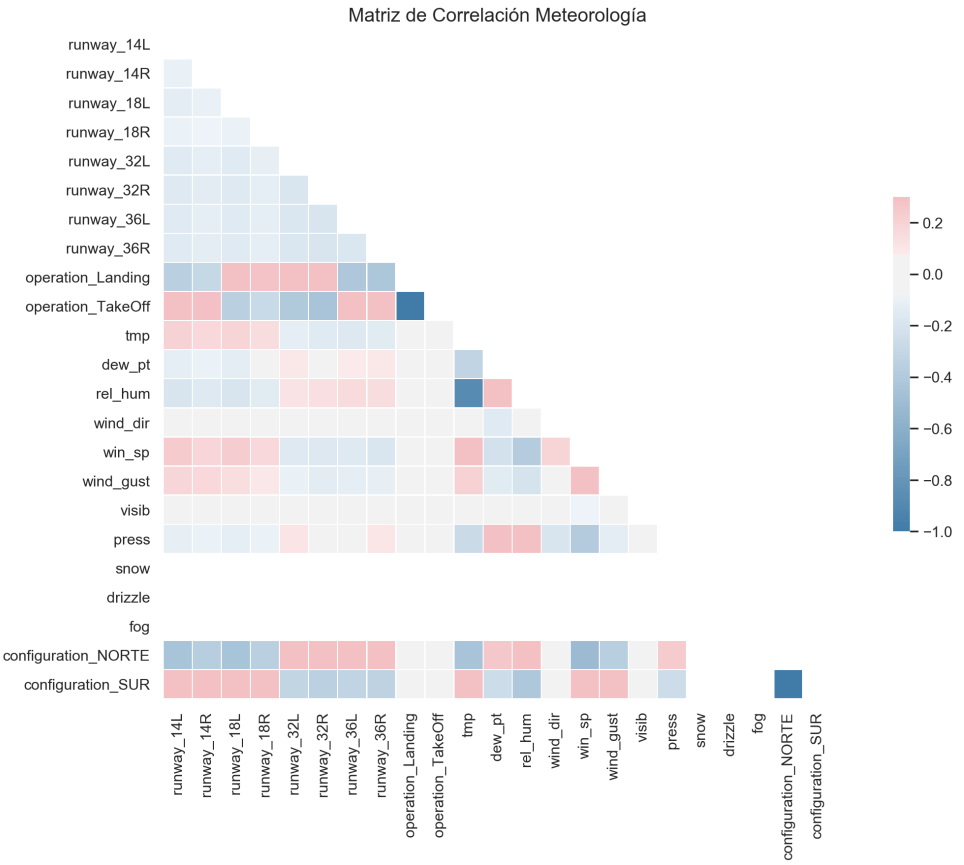


Figura 6.7: Matriz de Correlación para los atributos en función de factores meteorológicos.

Capítulo 7

Construcción del modelo

7.1. Elección inicial de modelos

Una vez preprocesados y refinados los datos que se van a emplear para entrenar y evaluar los modelos de aprendizaje automático, el siguiente paso consiste en realizar una prueba inicial con los principales modelos de aprendizaje automático destinados a tareas de clasificación. El objetivo de esta prueba inicial es aplicar estos modelos, con sus parámetros por defecto, al problema de la configuración de las pistas, para elegir el que mejor comportamiento presente. Después, calcularemos los valores más adecuados de los hiperparámetros de este modelo para optimizar su eficiencia. Para ello, se van a analizar los principales modelos de aprendizaje automático enfocados en la clasificación, además de los que han sido utilizados en anteriores propuestas (ver Sección 5.1). Los algoritmos que se van a utilizar son: K-Vecinos Más Cercanos, Regresión Logística, Bosques Aleatorios y Redes Neuronales.

Las pruebas iniciales de estos modelos se llevarán a cabo con 4 subconjuntos de datos distintos (se utilizarán todos los registros y se variarán las características) para determinar que conjunto de características se comporta mejor y obtiene mejores resultados con respecto a las demás. Estos subconjunto de datos son:

1) **Todas las características y variable de destino hot encoding (FULL):** Esta distribución de datos se basa en utilizar como predictores de la configuración de las pistas todas las características con las que cuenta el dataset, independientemente de la relación o influencia de estas con la propia configuración. La variable de destino que se utilizará para esta distribución es la configuración de la pista en su representación *hot encoding*, es decir, en lugar de utilizar como destino una única columna, se utilizarán las que se correspondan a la codificación en caliente de la configuración.

2) **Todas las características y variable de destino etiqueta de clase (FNORM):** Esta distribución es muy similar a la distribución anterior con la diferencia de que, en esta, la característica que utilizamos como destino se corresponde al valor de la configuración en su estado como etiqueta de clase. De esta forma se espera observar si existen grandes diferencias en el resultado obtenido utilizando como destino una característica en una representación u otra.

3) **Todas las características relacionadas y variable de destino hot enconding (RELATED):** Para esta distribución, se han utilizado únicamente aquellas características que

de algún modo tengan relación e influencia en la decisión de elegir una configuración u otra. Esta relación analizó en profundidad en la Sección 6.4, en la que se observó como no todas las características de las que se dispone influyen en la elección de una configuración u otra. Al igual que la distribución (FULL), la variable de destino empleada será la configuración de las pistas en su representación hot encoding.

4) **Todas las características relacionadas y variable de destino etiqueta de clase (RNORM):** Esta es la última distribución elegida para comprobar la eficacia de los modelos. Al igual que la anterior, las características utilizadas para calcular la configuración serán aquellas que tengan relación con la configuración de las pistas, y la variable de destino será la configuración de las pistas en su representación como etiqueta de clase.

Para llevar a cabo este análisis inicial de los modelos, se ha realizado una división estratificada sobre el conjunto de datos, de tal forma que el 80 % de estos (587.057 registros) serán utilizados para el entrenamiento del modelo (*Train Set*), y el 20 % restante (146.765 registros) se empleará para la evaluación (*Test Set*). En este caso en concreto no se va a utilizar un conjunto de validación (*Vaidation Set*), debido a que ninguno de los modelos de aprendizaje automático que se van a emplear hacen uso de este subconjunto de datos, por lo que prescindir de él permitirá disponer de un mayor conjunto de entrenamiento y de prueba.

En el caso de las métricas, la comparativa entre los modelos de aprendizaje automático se realizará comparando su precisión (cuántos ejemplos ha clasificado correctamente del total de ejemplos positivos) sobre el conjunto de datos de prueba, siendo mejor el modelo que mayor valor de precisión arroje.

7.1.1. Bosques aleatorios

Como ya vimos en la Sección 5.2, los bosques aleatorios son un tipo especial de modelo de aprendizaje automático denominado *ensemble*, cuya característica principal reside en que estos se encuentran constituidos por modelos más simples y cuyo resultado se obtiene por votación entre los modelos que conforman este *ensemble*, siendo la salida final la clase más votada entre todos los modelos del *ensemble*. En el caso de los bosques aleatorios, se encuentran constituidos por árboles de decisión, cuya configuración e hiperparámetros son iguales para cada uno de ellos, pero han sido entrenados con conjuntos de datos diferentes. La evaluación de este modelo se ha realizado con la siguiente configuración de hiperparámetros:

1) **Número de estimadores:** Representa el número de árboles que conforman el bosque. Su valor será variable para ver la evolución del modelo con distintos tamaños del bosque.

2) **Profundidad:** Representa el número de niveles de profundidad máximos que va a tener cada uno de los árboles. El valor utilizado es: *NULL* (sin límite).

3) **Mínimo número de ejemplos para dividir:** Representa el número mínimo de muestras antes de realizar una división en el nodo que la forman. El valor utilizado es *2*.

4) **Mínimo de ejemplos por hoja:** Representa el número mínimo de muestras en un nodo hoja antes de realizar una división en dicho nodo. El valor utilizado es *1*.

La Tabla 7.1 muestra los mejores resultados que se han obtenido con el modelo de bosques aleatorios para cada una de las distribuciones de datos:

	Número de árboles en el bosque					
Tipo	**20**	**50**	**100**	**200**	**300**	**500**
FULL	0,99671	0,99671	0,99665	0,99665	0,99665	0,996658
FNORM	0,99675	0,99666	0,99664	0,99666	0,99666	0,996680
RELATED	0,99654	0,99656	0,99651	0,99648	0,99647	0,996481
RNORM	0,99654	0,99655	0,99647	0,99650	0,99647	0,996468

Tabla 7.1: Precisión del modelo Bosques Aleatorios para cada distribución de datos.

Figura 7.1: Precisión del modelo bosques aleatorios en función del número de estimadores.

Como se puede ver en la Figura 7.1, la precisión obtenida con los bosques aleatorios es muy buena (ronda el 99 %) y se mantiene prácticamente constante independientemente del número de estimadores (árboles) que conforman el bosque. Esto hace que descartemos la utilización de un número elevado de estimadores para el modelo final, ya que el coste computacional y el tiempo asociado a un aumento en el número de estimadores, es elevado para la mejora final real que se va a obtener (la cual es del orden de 10^{-3}), por lo que no merece la pena pagar ese coste. Otra característica a destacar del rendimiento es que, las distribuciones de datos basadas en las características que tienen relación con la configuración de las pistas funcionan ligeramente peor que en las que se utilizan todos los datos. Esto puede deberse a que el modelo es capaz de encontrar en estos datos con menor grado de relación información relevante que le ayuda a calcular la configuración, a pesar de que estos no guarden una relación directa con la configuración.

7.1.2. K-Vecinos Más Cercanos

Para el caso del modelo de K-Vecinos Más Cercanos, se ha optado por observar la evolución en la eficacia del modelo con respecto al número de vecinos que se utilizan para realizar la clasificación, siendo este el parámetro fundamental a la hora de establecer la clasificación. La

configuración de los hiperparámetros utilizada para evaluar el rendimiento del modelo de K-Vecinos Más Cercanos es la siguiente:

1) **Número de vecinos:** Representa el número de vecinos que se utilizarán para establecer la clasificación. Su valor será variable para observar la evolución del modelo con distinto número de vecinos.

2) **Pesos:** Los pesos representan el impacto o valor que se asocian a cada uno de los vecinos en función de su cercanía. Puede ser uniforme (la importancia de cada uno de los vecinos es la misma) o por distancia (los vecinos que se encuentren mas próximos tendrán un mayor peso que los que se encuentren más alejados). El valor que se utilizará es *uniforme* (todos los vecinos tienen la misma importancia).

3) **Distancia:** Representa la función de distancia que se utilizará para calcular la proximidad entre los vecinos. La distancia que se utilizará es la Euclídea (*distancia de Minkowski con* $p = 2$), la cual viene dada por la siguiente expresión:

$$D(X, Y) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{\frac{1}{p}}$$

La Tabla 7.2 muestra los mejores resultados obtenidos con el modelo de K-Vecinos Más Cercanos para cada una de las distribuciones de datos:

	Número de vecinos						
Tipo	**1**	**3**	**5**	**7**	**9**	**11**	**13**
FULL	0,91825	0,92386	0,92374	0,92352	0,92210	0,92066	0,91909
FNORM	0,91769	0,92383	0,92401	0,92383	0,92251	0,92088	0,91944
RELATED	0,91701	0,92323	0,92371	0,92281	0,92201	0,92033	0,91844
RNORM	0,91750	0,92295	0,92411	0,92300	0,92177	0,92051	0,91917

Tabla 7.2: Precisión del modelo K-Vecinos Más Cercanos para cada distribución de datos.

Finalmente, como se puede observar en la Figura 7.2, existe un máximo en cuanto a la precisión obtenida por el modelo para un valor de (3-9 vecinos). A partir de esa cantidad, la precisión comienza a disminuir considerablemente a medida que se va aumentando la cantidad de vecinos. Además de esto, a pesar de que todas las distribuciones de datos se comportan de una forma muy similar a medida que aumenta el número de vecinos, en el intervalo máximo de la precisión del modelo (entre 3 y 9 vecinos), las distribuciones que mejor se comportan son aquellas que utilizan todas las características de los datos independientemente del impacto de estas sobre la variable a clasificar. Nuevamente esto puede deberse a que el modelo de K-Vecinos sea capaz de obtener información sobre estas que le ayude a la hora de realizar la clasificación, mejorando así la precisión global.

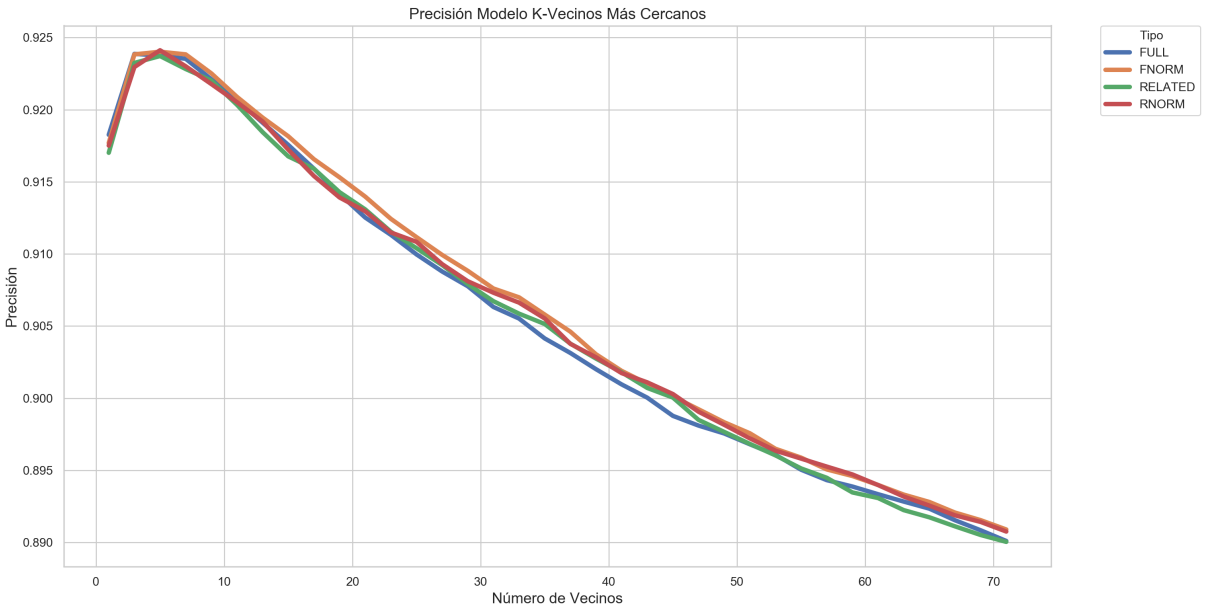


Figura 7.2: Precisión del modelo K-Vecinos Más Cercanos en función del número de vecinos.

7.1.3. Regresión Logística

En el caso de la regresión logística, se ha realizado únicamente el análisis con la distribución de datos *FNORM* y *RNORM* ya que, este tipo de modelos no acepta más de una característica como variable de destino con lo que hace imposible su utilización con las distribuciones *FULL* y *RELATED*.

La configuración de los hiperparámetros utilizada para evaluar el rendimiento del modelo de regresión logística es el siguiente:

1) **Algoritmo de Resolución:** Algoritmo que se utilizará para optimizar el modelo y minimizar su error. Su valor será variable para poder observar la evolución del modelo para cada algoritmo de optimización.

2) **Penalización:** Se utiliza para determinar la regularización que se utilizará a la hora de establecer la penalización. El valor utilizado es *l2*.

La Tabla 7.3 muestra los mejores resultados obtenidos con el modelo de regresión logística para las distintas distribuciones de datos:

Tipo	Algoritmo de Resolución	Precisión
FNORM	liblinear	0.79583
FNORM	newton-cg	0.79459
RNORM	sag	0.79108
RNORM	newton-cg	0.78353

Tabla 7.3: Precisión del modelo de Regresión Logística para cada distribución de datos.

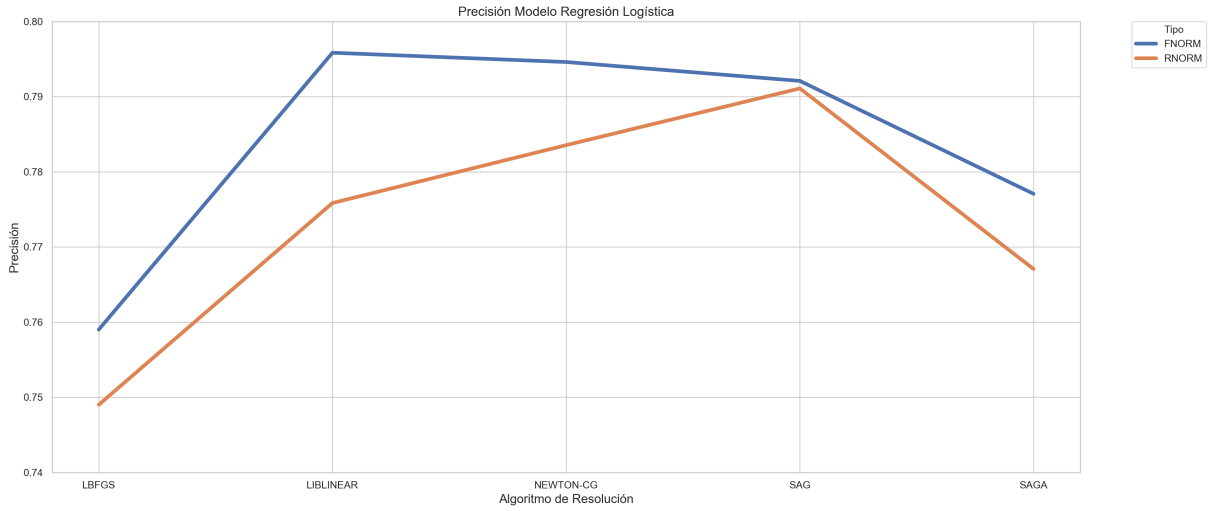


Figura 7.3: Precisión del modelo de Regresión Logística en función del algoritmo de resolución.

Seguidamente, como se puede apreciar en la figura (ver Figura 7.3), la precisión obtenida con el modelo de regresión logística es bajísima en comparación con la del resto de modelos (aproximadamente un 12-15 % inferior). Esto nos lleva a pensar que es posible que la combinación de los hiperparámetros no sea la más óptima y que con un correcto ajuste de los mismos se consiga mejorar la precisión. Del mismo modo (y al igual que ocurre con el resto de modelos), sigue obteniendo mejores resultados la distribución de datos en la que se contemplan todas las características disponibles. Para el caso de la regresión logística, la brecha entre una distribución y otra alcanza su valor máximo, siendo esta de aproximadamente un 1-2 % (frente a la brecha de $1 \cdot 10^{-3}$ existente en el resto de los modelos anteriores.

7.1.4. Redes Neuronales

El último modelo que ha sido probado para resolver este problema son las redes neuronales. Este modelo tiene un gran número de hiperparámetros a modificar por lo que, en este caso, se va a realizar el análisis con respecto a dos parámetros básicos en un modelo basado en redes neuronales: la función de activación y el número de épocas. El objetivo de esto consiste en elegir tanto la mejor función de activación como el mejor número de épocas, para posteriormente analizar los resultados con distintas configuraciones de capas y neuronas, fijando en ellas estos hiperparámetros y facilitando la obtención de la configuración más óptima. Por lo tanto la configuración de los hiperparámetros utilizada para evaluar este modelo es la siguiente:

1) **Configuración de la red:** Representa el número de capas y de neuronas que conforman la red neuronal. La configuración utilizada es *(100)* (1 capa de 100 neuronas).

2) **Función de activación:** Tipo de función de activación que tendrá cada una de las neuronas de la red. Este parámetro será variable para poder así obtener la función que obtenga mejores resultados.

3) **Algoritmo de Resolución:** Algoritmo que se utilizará para optimizar el modelo y minimizar el error. El algoritmo de resolución utilizado es *Adam*.

4) **Factor de aprendizaje:** Representa qué rápido aprende la red. El valor utilizado es: $\alpha = 0{,}001$.

5) **Tipo de factor de aprendizaje:** Indica cómo se comporta el valor del factor de aprendizaje en cada una de las épocas durante el entrenamiento. El valor utilizado es: *Constante* (siempre es el mismo).

En la Tabla 7.4, se muestra el resultado obtenido para cada distribución de datos en función de la función de activación utilizada:

	Función de Activación			
Tipo	**Identidad**	**Logística**	**ReLU**	**TanH**
FULL	0,92556	0,76014	0,92581	0,75836
FNORM	0,92585	0,76131	0,92379	0,76175
RELATED	0,87084	0,75990	0,79001	0,76066
RNORM	0,91543	0,76119	0,91490	0,76187

Tabla 7.4: Precisión del modelo de Redes Neuronales para cada distribución de datos en función de la activación.

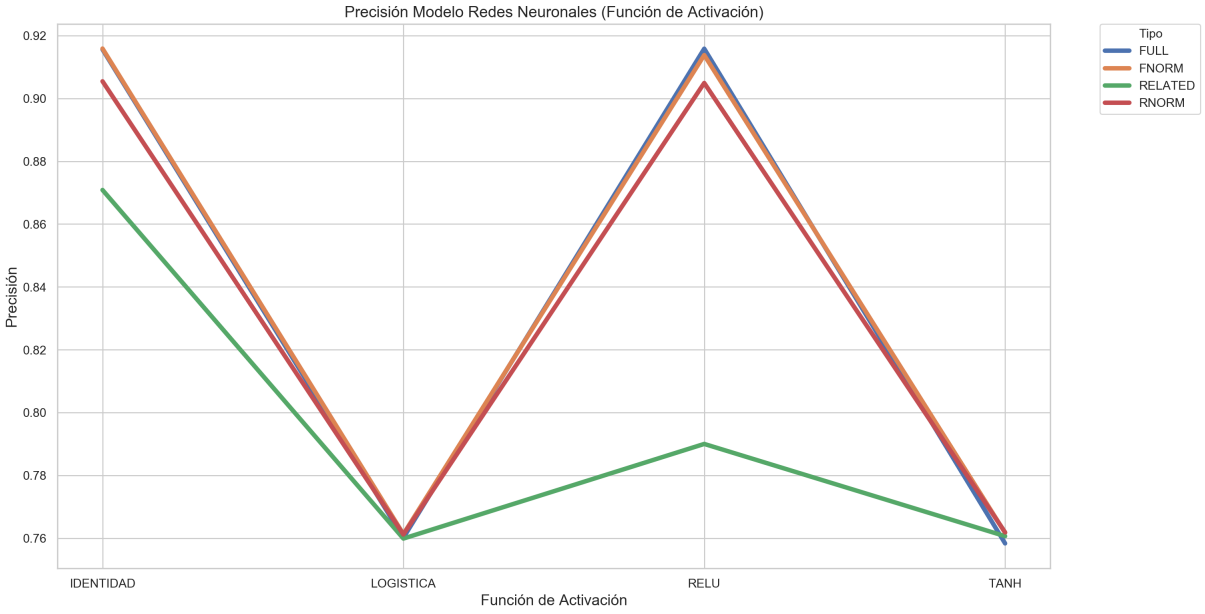


Figura 7.4: Precisión del modelo de Redes Neuronales en función de la activación.

Además de esto, como se puede apreciar en la Figura 7.4, las funciones de activación que obtienen unos mejores resultados con respecto al resto son la *función identidad* y la *función ReLU*, siendo su precisión de casi un 92 %. Una vez hecho este análisis, vamos a observar cómo se comporta cada una de estas funciones para un número variable de épocas, analizando su comportamiento y decidiendo cuál de ellas será la que se utilice en el modelo final.

Como se puede observar en la Tabla 7.5, a medida que aumenta el número de épocas, lo hace también la precisión, independientemente del tipo de función de activación que se utilice. No

obstante, la diferencia que existe en la precisión utilizando una función u otra es muy notoria, ya que, en el caso de las funciones *ReLU* e *Identidad*, la precisión es muy alta (un 92 %) siendo esta configuración inicial la que mejores resultados ha obtenido junto con el modelo basado en bosques aleatorios. Asimismo, las funciones *Logística* y *Tangente Hiperbólica (TanH)* han obtenido los peores resultados de todas las pruebas realizadas por lo que su utilización queda descartada. Finalmente, cabe destacar la mejora que existe en función de la distribución de datos utilizada, y es que, aquellas distribuciones en las que se utilizan todas las características de los datos obtienen mejores resultados que el resto.

	Identidad				**Logística**			
Tipo/Épocas	**20**	**50**	**100**	**150**	**20**	**50**	**100**	**150**
FULL	0,82148	0,91527	0,91556	0,91529	0,76012	0,76023	0,76014	0,76236
FNORM	0,90755	0,91335	0,91585	0,91581	0,76198	0,76093	0,76131	0,76046
RELATED	0,48913	0,47099	0,87084	0,91535	0,76067	0,76158	0,75990	0,76162
RNORM	0,89404	0,91498	0,90543	0,91563	0,76160	0,76131	0,76119	0,75993

	ReLU				**TanH**			
Tipo/Épocas	**20**	**50**	**100**	**150**	**20**	**50**	**100**	**150**
FULL	0,51155	0,91250	0,91581	0,91557	0,75992	0,76033	0,75836	0,76174
FNORM	0,84661	0,91410	0,91379	0,91570	0,76136	0,75974	0,76175	0,75999
RELATED	0,90212	0,91162	0,79001	0,91521	0,75919	0,75955	0,76066	0,76112
RNORM	0,76789	0,91340	0,90490	0,83167	0,76023	0,76024	0,76187	0,75870

Tabla 7.5: Evolución precisión Redes Neuronales en función de la distribución de datos, el número de épocas y la función de activación.

Conclusiones

A la vista de los resultados obtenidos en un análisis inicial de los principales algoritmos de aprendizaje automático y utilizando distintas distribuciones de datos, podemos afirmar con total certeza que las distribuciones de datos que mejores resultados obtienen, son las que se encuentran constituidas por todos las características que conforman el dataset. Esta diferencia es notoria en algunos modelos como en el caso de las redes neuronales, o puede ser menos pronunciada como en el caso del modelo basado en bosques aleatorios. No obstante, esta diferencia existe y ha quedado demostrada a lo largo de las pruebas realizadas (ver Figura 7.5) por lo que, el proceso de optimización de los modelos se realizará únicamente con las distribuciones de datos en las que contamos con todas las características. En relación a lo anterior y viendo que la diferencia en cuanto a la precisión obtenida entre ambas distribuciones de datos es ínfima, de las dos distribuciones de datos que cuentan con todas las características (*FULL* y *FNORM*), se va a optar por optimizar los modelos con esta última. Esto se debe que la variable de destino de esta es una única columna con los valores normalizados (en lugar de tener más de una columna como destino), eliminando así la limitación existente con algunos modelos que no admiten más de una columna como destino, consiguiendo así que la distribución de datos sea "genérica" y se pueda aplicar cualquier modelo sobre la misma, evitando posibles problemas de compatibilidad

que puedan surgir a lo largo del proceso de optimización.

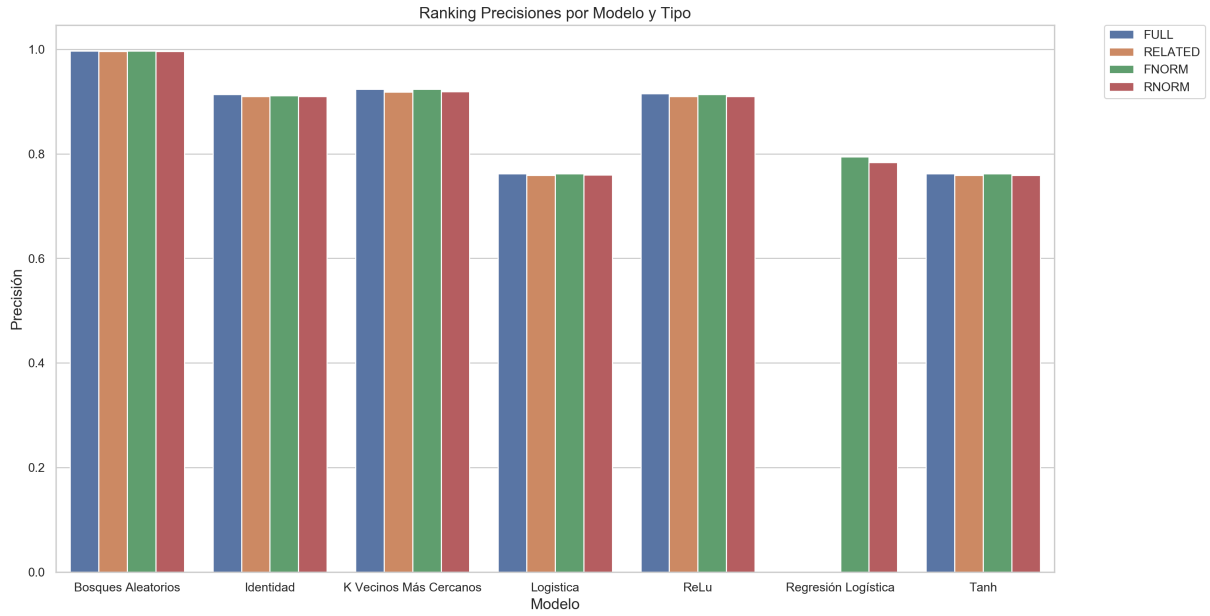


Figura 7.5: Precisión global de los modelos de aprendizaje automáticos analizados.

7.2. Mejorando los modelos iniciales

El siguiente paso una vez realizado un análisis exploratorio inicial de los resultados obtenidos utilizando los principales algoritmos de clasificación, es realizar un proceso de refinado y de mejora sobre estos, el cual consiste en modificar los distintos hiperparámetros disponibles en los modelos con la finalidad de alcanzar mejores resultados que los obtenidos en la sección anterior.

La configuración de partida elegida para cada modelo (basándonos en los resultados anteriores) y sobre la cual se va a trabajar para mejorarla es la siguiente:

Modelo	Parámetros base seleccionados	Precisión
Bosques Aleatorios	- Número de Árboles: 200	0,996680
K-Vecinos Más Cercanos	- Número de Vecinos: 5	0,924015
Regresión Logística	- Algoritmo de Resolución: Liblinear	0,795830
Redes Neuronales	- Función de Activación: ReLU - Épocas: 100-150	0,915707

Tabla 7.6: Configuración base óptima de los modelos a mejorar.

Al igual que ocurría con la evaluación inicial de los modelos de aprendizaje automático, se va a realizar una división estratificada del conjunto de datos final, de tal forma que el 80 % de estos se utilizarán para entrenar los modelos sobre los que se están realizando modificaciones, y el 20 % restante se utilizará para probar la eficacia de los mismos.

Del mismo modo, la métrica que se va a utilizar para comparar los modelos será la precisión,

la cuál simboliza el porcentaje de clasificaciones correctas realizadas sobre el total de ejemplos de correctos. Se considerarán mejores aquellos modelos con un mayor valor de precisión.

7.2.1. Bosques aleatorios

Para el caso de los bosques aleatorios, la precisión inicial obtenida es muy alta (es prácitcamente de un 100 %), no obstante se va a comprobar si es posible aumetar dicha precisión. Para ellos los parámetros a modificar son lo siguientes:

1) **Profundidad:** Representa el número de niveles de profundidad máximos que va a tener cada uno de los árboles. El valor de la profundidad tomará valores pertenecientes al intervalo $[1, 41]$ siendo 1 el valor mínimo que esta puede tomar y 41 el máximo, ya que contamos con 41 características y no puede haber más niveles de profundidad que características.

   A continuación se muestran los resultados obtenidos al variar la profundidad de los árboles que conforman el bosque aleatorio (ver Tabla 7.7 y ver Figura 7.6).

Profundidad	**Precisión**
1	0,75914
3	0,83597
5	0,89582
7	0,90982
15	0,97889
18	0,99358
30	0,99558
41	0,99883

Tabla 7.7: Precisión Bosque Aleatorio (200 árboles) en función de la profundidad del árbol.

   Como se puede observar, la precisión del modelo mejora considerablemente a medida que se aumenta la profundidad del árbol, coincidiendo el valor máximo de la profundidad con el valor máximo de precisión. No obstante, a partir de la profundidad 18, la precisión apenas aumenta, por lo que duplicar la profundidad del árbol para mejorar la precisión en unas pocas milésimas no merece la pena computacionalmente hablando. Es por esto por lo que la elección para la profundidad va a ser: **profundidad = 18**

2) **Mínimo número de ejemplos para dividir:** Representa el número mínimo de muestras antes de realizar una división en el nodo en el que se encuentran. Este valor variará entre [0,1, 1,0], que expresado matemáticamente es equivalente a:

$$\lceil \text{mínimo de ejemplos para dividir} \cdot \text{número total de ejemplos} \rceil$$

   A continuación se muestran los resultados obtenidos al variar el número de ejemplos antes de realizar una división en el nodo interno que lo forman (ver Tabla 7.8 y ver Figura 7.7).

   Para el caso del mínimo número de ejemplos para dividir, la precisión del modelo disminuye a gran velocidad a medida que aumentamos el número de ejemplos necesarios para realizar una división en un nodo. Esto se debe a que al "esperar" más para realizar dicha división, el

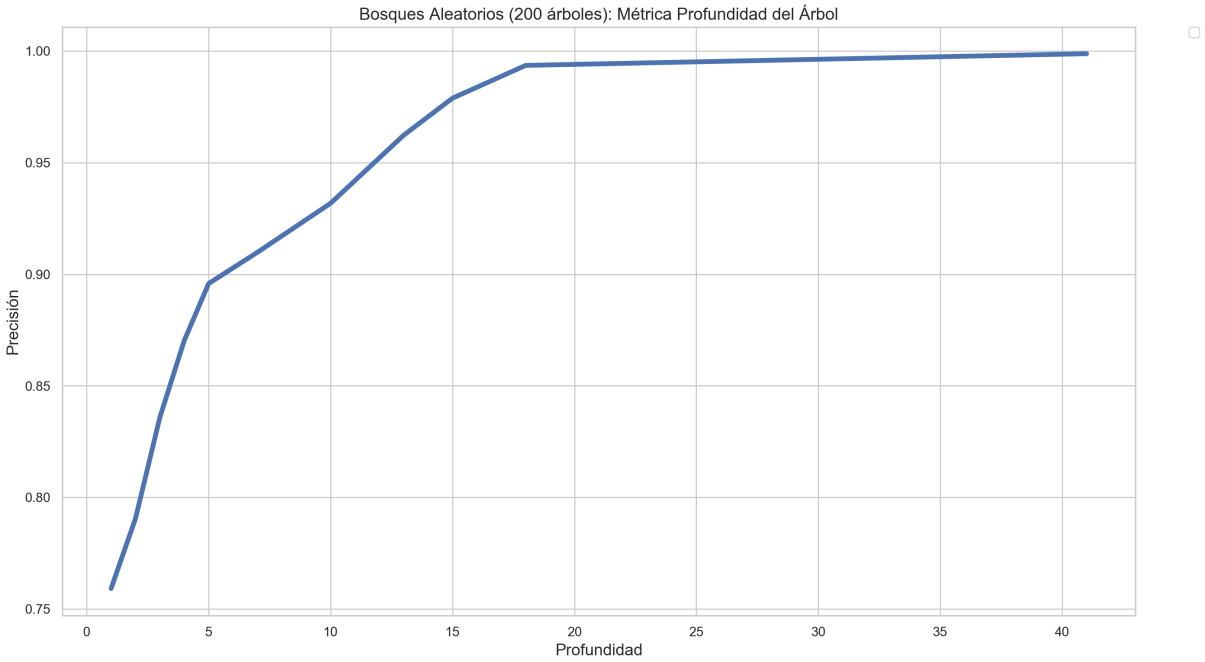


Figura 7.6: Precisión Bosque Aleatorio (200 árboles) en función de la profundidad del árbol.

Mínimo Número de ejemplos	**Precisión**
0,1	0,88824
0,3	0,83329
0,5	0,75914
0,7	0,75914
0,9	0,75914
1,0	0,75914

Tabla 7.8: Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir una característica.

desglose en características también se retrasa por lo que es más probable que se clasifiquen erróneamente muestras que seguramente se habrían clasificado bien si se hubiese realizado antes esta división. Por tanto el valor que se tomará para esta característica es: **mínimo numero de ejemplos para dividir = 2**

3) **Mínimo de ejemplos por hoja:** Representa el número mínimo de muestras en un nodo hoja antes de realizar una división en dicho nodo. Este valor variará entre [0,1, 0,5] que expresado matemáticamente es equivalente a:

$$\lceil \text{mínimo de ejemplos por hoja} \cdot \text{número total de ejemplos} \rceil$$

A continuación se muestran los resultados obtenidos al variar el número de ejemplos antes de realizar una división en el nodo hoja (ver Tabla 7.9 y ver Figura 7.8).

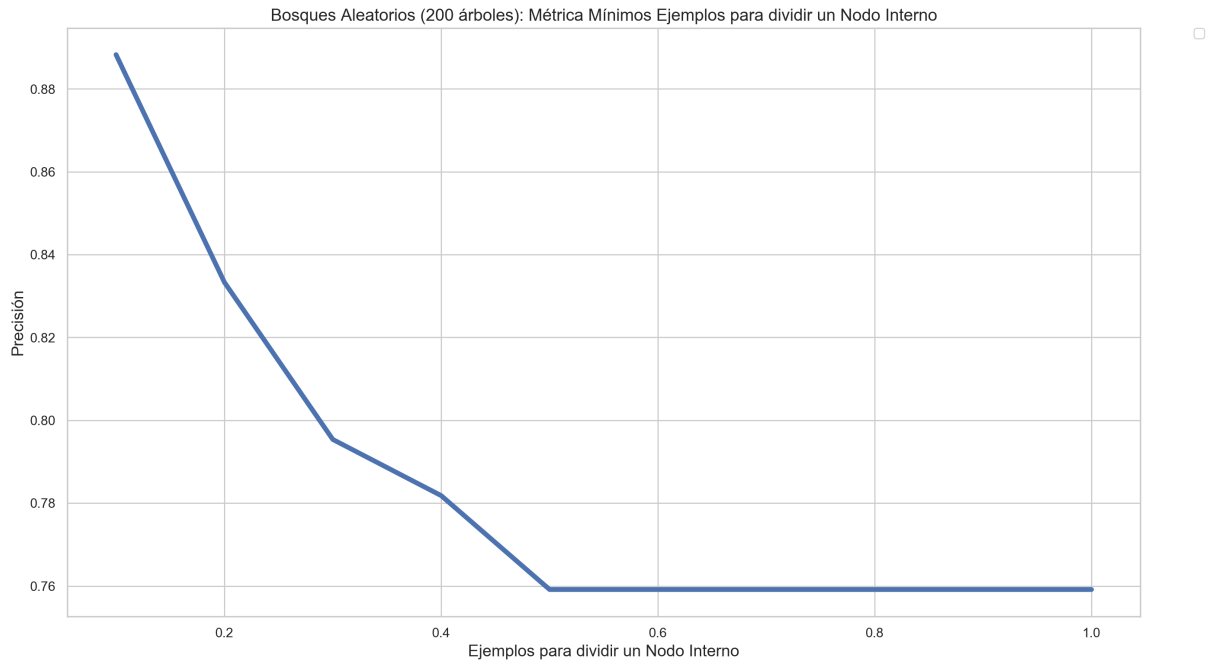


Figura 7.7: Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir una característica.

Mínimo número de ejemplos por hoja	**Precisión**
0,1	0,75914
0,2	0,75914
0,3	0,75914
0,4	0,75914
0,5	0,75914

Tabla 7.9: Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir un nodo hoja.

Para el caso de la división de un nodo hoja en función de los ejemplos que tenga, se puede observar que es totalmente indiferente ya que la precisión se mantiene constante independientemente del resultado. Esto significa que para este caso, este hiperparámetro no tiene influencia alguna sobre el resultado final, por lo que será descartado a la hora de optimizar el modelo, y se seleccionará el valor que tiene por defecto. Por tanto el valor que se tomará para esta característica es: **mínimo número de ejemplos para dividir un nodo hoja = 1**

Finalmente, y de acuerdo a los resultados obtenidos en el análisis tras realizar las diferentes modificaciones en los hiperparámetros de los árboles de decisión que conforman el bosque aleatorio, el modelo final que se utilizará tiene la siguiente configuración (ver Tabla 7.10):

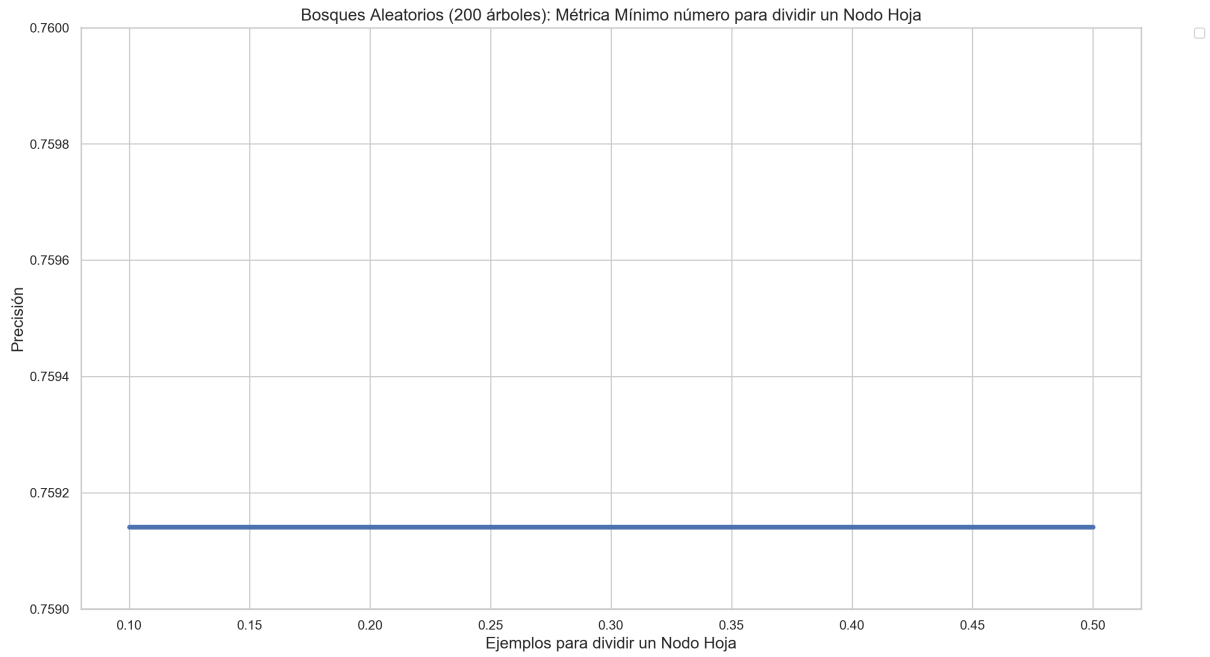


Figura 7.8: Precisión Bosque Aleatorio (200 árboles) en función del mínimo número de ejemplos para dividir un nodo hoja.

Hiperparámetro	Valor antiguo	Valor Nuevo
Número de árboles	200	200
Profundidad	Sin límite	18
Mínimo número de ejemplos para dividir	2	2
Mínimo de ejemplos por hoja	1	1

Tabla 7.10: Modificaciones realizadas en los hiperparámetros del bosque aleatorio.

7.2.2. K-Vecinos Más Cercanos

En el caso del algoritmo de K-Vecinos Más Cercanos, los resultados obtenidos rondan el 92 % de precisión. A pesar de esto, se van a realizar las siguientes modificaciones en los hiperparámetros:

1) **Pesos:** Los pesos representan el impacto o valor que se asocian a cada uno de los vecinos en función de su cercanía. Puede ser uniforme (la importancia de cada uno de los vecinos es la misma) o por distancia (los vecinos que se encuentren mas próximos tendrán un mayor peso que los que se encuentren más alejados). Los criterios para determinar el valor de los pesos que se han utilizado ha sido *uniforme* (todos los vecinos tienen la misma influencia independientemente de su distancia) y *distancia* (los vecinos más próximos tendrán una influencia mayor que aquellos que se encuentren más alejados).

   A continuación se muestran los resultados obtenidos al variar valor de los pesos a la hora de ponderar la influencia de cada vecino en función de su cercanía (ver Tabla 7.11 y ver Figura 7.9).

Asignación de los pesos	Precisión
UNIFORME	0,93028
DISTANCIA	0,93355

Tabla 7.11: Precisión K-Vecinos Más Cercanos (5 vecinos) en función de la asignación de pesos.

Para el caso de la forma en la que se asignan los pesos a los vecinos en función de la proximidad, se puede observar que no existen diferencias notorias entre ambas, siendo esta del orden de $1 \cdot 10^{-3}$. Esto implica que, en este caso, la diferencia entre utilizar uno u otro no afecta al rendimiento final del modelo. No obstante, la elección de la forma en la que se establecen los pesos a los vecinos es: **distancia**. (en función de la distancia a la que se encuentren).

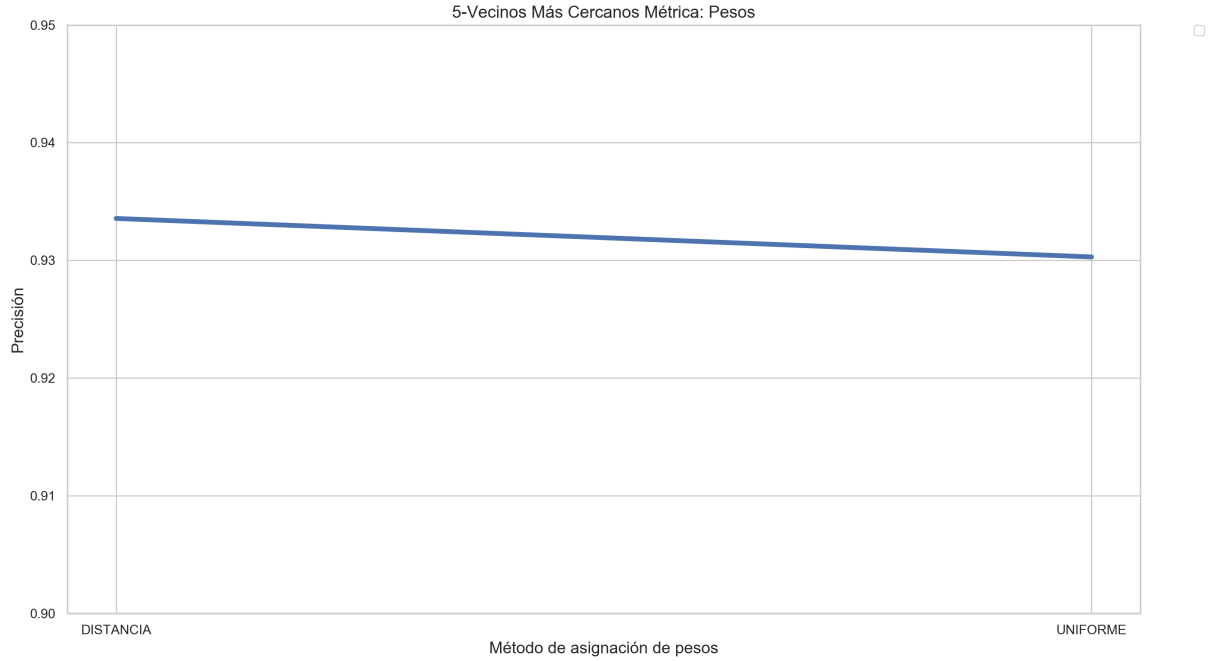


Figura 7.9: Precisión K-Vecinos Más Cercanos (5 vecinos) en función de la asignación de pesos.

2) **Distancia:** Representa la función de distancia que se utilizará para calcular la proximidad entre los vecinos. Las distancias utilizadas para probar el rendimiento son: *distancia euclídea*, *distancia de Manhattan* y la *distancia de Chebyshev*.

   A continuación se muestran los resultados obtenidos al variar valor de la función de distancia empleada (ver Tabla 7.12 y ver Figura 7.10).

   Para el caso de la función con la que se calcula la distancia a otros vecinos, podemos ver como existen grandes diferencias entre las utilizadas, siendo la distancia de Manhattan y la Euclídea las mejores de todas, siendo la de Manhattan muy superior a la Euclídea (más de un 2 % de diferencia entre ambas). Esto se debe a que la distancia de Manhattan representa el tipo de distancia que hay: positiva (en el caso de que sea mayor que 0) o negativa (en caso contrario). Este detalle es fundamental, ya que es probable que, para esta casuística, el saber si un vecino se encuentra a una distancia X positiva o una distancia X negativa hace que

Distancia	Precisión
Euclídea	0,93038
Manhattan	0,95282
Chebyshev	0,91405

Tabla 7.12: Precisión K-Vecinos Más Cercanos (5 vecinos) en función del algoritmo para obtener la distancia entre los vecinos.

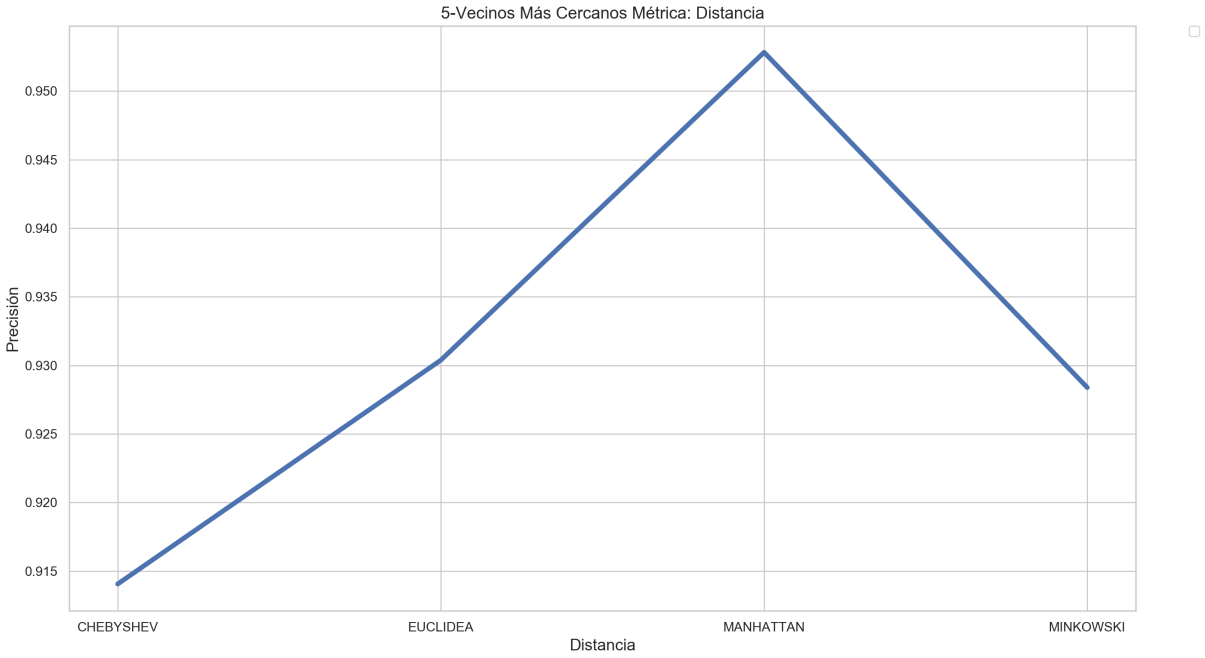


Figura 7.10: Precisión K-Vecinos Más Cercanos (5 vecinos) en función del algoritmo para obtener la distancia entre los vecinos.

muchos ejemplos que antes se encontraban en la frontera, se clasifiquen correctamente. Por tanto la elección del algoritmo que determine la distancia entre dos vecinos es: **Manhattan**.

De acuerdo a los resultados obtenidos en el análisis tras realizar las diferentes modificaciones en los hiperparámetros que conforman el modelo del K-Vecinos Más Cercanos, la configuración del modelo final será la siguiente (ver Tabla 7.13):

Hiperparámetro	Valor Antiguo	Valor Nuevo
Número de Vecinos	5	5
Algoritmo asignación de pesos	Uniforme	Distancia
Distancia	Euclídea	Manhattan

Tabla 7.13: Modificaciones realizadas en los hiperparámetros del K-Vecinos Más Cercanos.

7.2.3. Regresión Logística

El modelo de regresión logística, fue el que peor precisión obtuvo (79,5 %) durante el fase inicial de exploración de algoritmos de clasificación (ver Sección 7.1). No obstante, se van a realizar distintas modificaciones sobre los parámetros del mismo, con el objetivo de poder mejorar dicho algoritmo y ver si es posible que pueda estar al mismo nivel que los anteriores. Los parámetros modificados en el algoritmo de regresión logística son:

1) **Algoritmo de Resolución:** Algoritmo que se utilizará para optimizar el modelo y minimizar su error. Los valores que se han utilizado son: *Newton-cg*, *Lbfgs*, *Liblinear Sag* y *Saga*.

2) **Penalización:** Se utiliza para determinar la regularización que se utilizará a la hora de establecer la penalización. Los valores que se han utilizado son: *L1*, *L2* y *Elasticnet*.

3) **Parámetro C:** Representa el valor inverso de la fuerza de regularización. Cuanto más pequeño sea mayor será la regularización que se aplique.

A continuación se muestran los resultados obtenidos al variar de forma simultánea cada cada uno de los hiperparámetros del algoritmo de regresión logística (ver Figura 7.11).

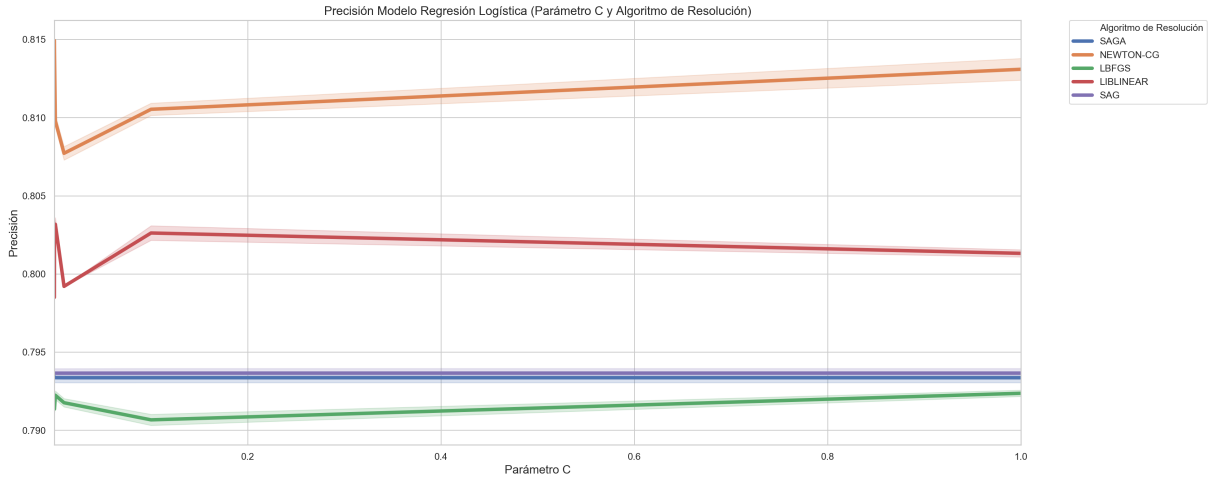


Figura 7.11: Precisión Regresión Logística en función de la penalización, el algoritmo de resolución y el parámetro C.

Como se puede observar, el algoritmo de resolución que mejor resultados arroja es el *Newton-cg* con un valor del parámetro C muy pequeño ($1 \cdot 10^{-5}$) cuyo porcentaje ronda el 81 % (un aumento del 2 % con respecto al modelo de regresión logística anterior), (ver Tabla 7.14). Para el resto de algoritmos, observamos que existen diferencias entre ellos ya que para el *saga*, *sag* y *lblfs* apenas se consigue pasar el 80 %, manteniéndose prácticamente en el mismo intervalo de precisión que en el modelo original. En el caso de la penalización, el valor utilizado ha sido *L2*, ya que es el único penalizador que admite *Newton-cg*. A pesar de esto, los resultados distan mucho de los obtenidos en anteriores modelos ya planteados y optimizados, por lo que viendo que este modelo no es capaz de llegar a los valores de precisión esperados, se descartará que pueda ser el modelo elegido para resolver este problema.

Penalizador	Algoritmo de Resolución	Valor de C	Precisión
L2	Newton-cg	$1 \cdot 10^{-5}$	0,81486
L2	Newton-cg	$1 \cdot 10^{-4}$	0,81379
L2	Newton-cg	$1 \cdot 10^{-3}$	0,80939
L2	Newton-cg	$1 \cdot 10^{-2}$	0,80728
L2	Newton-cg	0.1	0,81013
L2	Newton-cg	1.0	0,81239

Tabla 7.14: Precisión Regresión Logística para el penalizador L2 y el algoritmo de resolución Newton-cg en función del parámetro C.

7.2.4. Redes Neuronales

Finalmente, el último modelo de aprendizaje automático a optimizar es el modelo basado en redes neuronales. En primera instancia, el modelo basado en redes neuronales obtuvo una precisión de un 91 % utilizando la función de activación reLU. No obstante, la configuración de capas y neuronas utilizadas fue la predeterminada (1 capa con 100 neuronas), por lo que el objetivo principal consistirá en encontrar la mejor configuración de esta además de optimizar otros de los hiperparámetros que conforman la red. Por lo tanto, los parámetros a optimizar son:

1) **Configuración de capas y neuronas:** Este parámetro es uno de los más importantes a la hora de crear una red neuronal, ya que es el esqueleto de la red. Representa el número de capas internas que tendrá nuestra red así como el número de neuronas de cada una de estas capas.

   El procedimiento llevado a cabo para realizar esta elección ha consistido fijar un número máximo de capas (en este caso 4), de tal forma que se pueda observar la evolución que existe al aumentar las capas, y poder ver si existe una relación entre la precisión final obtenida y el número de capas internas. Paralelamente a esto, para cada número de capas internas, se ha ido incrementando el número de neuronas que la conforman para que, al igual que antes, observar si existe alguna tendencia o relación entre el número de capas y neuronas con la precisión final obtenida.

   Como se puede observar en la Figura 7.12, el número de capas que conforman la red es más relevante que el número de neuronas de la misma, ya que la diferencia en cuanto a la precisión final si únicamente variamos las neuronas hace que no afecte significativamente al resultado final (a partir de las 40 neuronas la diferencia de precisión no sufre apenas variaciones excepto para la configuración de 1 capa). Del mismo modo, en el caso del número de capas internas, sí que existen diferencias, ya que vemos que, a medida que aumentamos el número de estas, las variaciones en la precisión son cada vez más leves, llegando casi a no variar apenas (como ocurre para el caso de 4 capas ocultas). Por todas estas razones la elección en cuanto al **número de capas ocultas es 4**.

   En cuanto al número de neuronas, si únicamente nos fijamos en la precisión obtenida para 4 capas ocultas, el máximo se obtiene en el rango de [25, 40] neuronas por capa. Por lo que para elegir la configuración de neuronas, se han analizado distintas configuraciones que cumplen con la premisa de tener 4 capas ocultas con un valor de neuronas pertenecientes al

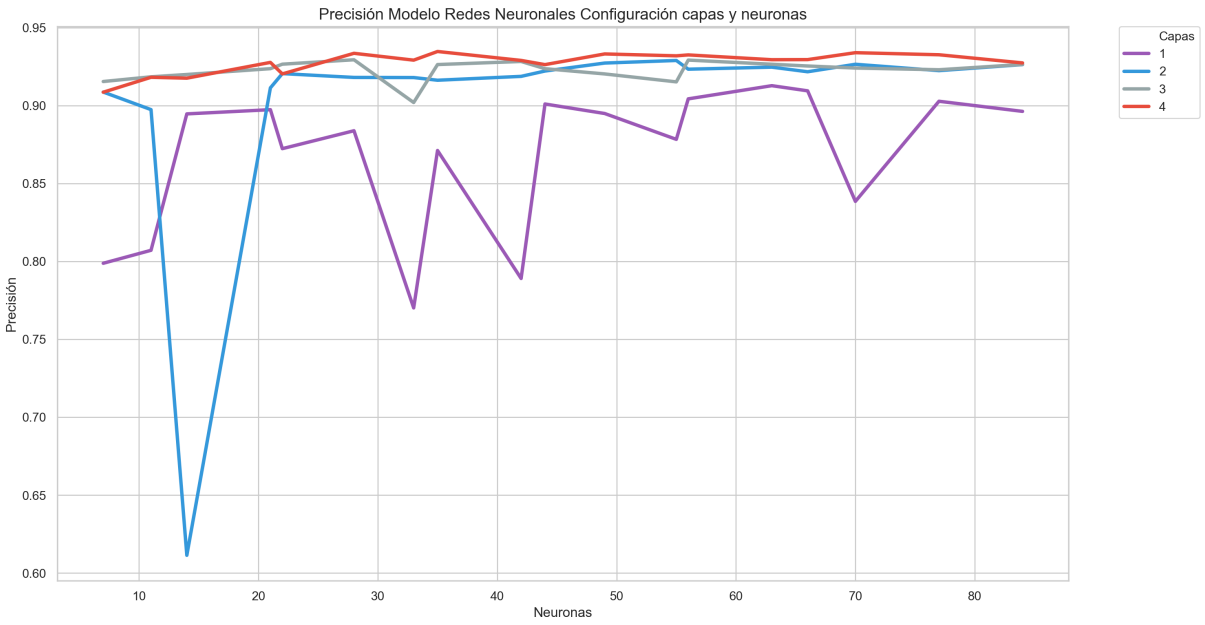


Figura 7.12: Precisión Redes Neuronales en función del número de capas y neuronas por capa.

intervalo [25, 40]. A continuación se observan los resultados obtenidos para cada una de estas configuraciones (ver Figura 7.13):

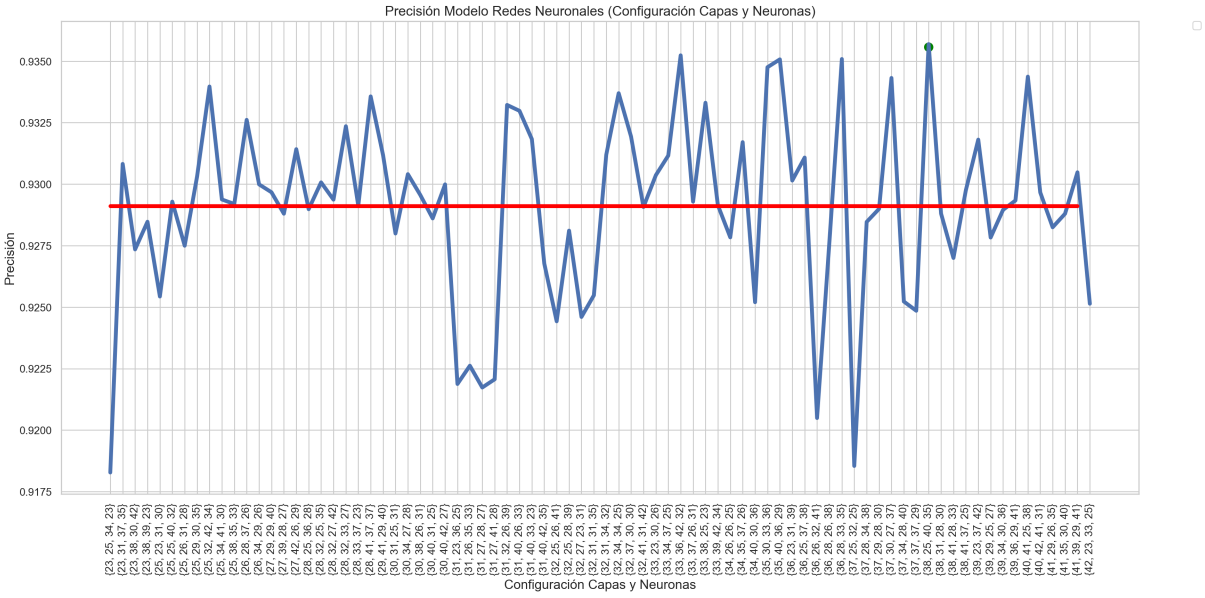


Figura 7.13: Precisión Redes Neuronales para 4 capas ocultas (el punto verde representa el mayor valor de la precisión obtenida).

Para este caso, vemos que la mayoría de las configuraciones se encuentran por encima de la media (línea roja), llegando incluso a alcanzar valores próximos al 94 % (cuando antes

eran inferiores al 92 %), lo que supone una mejora sustancial de la precisión final, ya que a estos niveles, el aumentar la precisión es muy complicado. En cuanto a la elección final de la configuración, se ha optado por seleccionar aquella con la que se ha obtenido una mayor precisión en la prueba anterior **(Capa 1: 38 neuronas, Capa 2: 25 neuronas, Capa 3: 40 neuronas y Capa 4: 35 neuronas)**.

2) **Factor de Aprendizaje:** El factor de aprendizaje representa la "velocidad" con la que una red neuronal aprende o, dicho de otro modo, es el valor que marca el tamaño del "salto" a la hora de movernos de un punto de la función a otro durante la búsqueda de la minimización del error. Un factor de aprendizaje muy pequeño hace que la red necesite de muchas operaciones hasta que consigue converger en dicho valor mínimo, y del mismo modo, un valor demasiado alto hace que los "saltos" que se dan entre los puntos de la función sean tan dispersos que nunca se llegue a alcanzar el valor mínimo de la función del error, por lo que la elección de este parámetro es de gran importancia para el correcto funcionamiento de la red. Para ello, se ha realizado el análisis probando distintos factores de aprendizaje, cuyos valores oscilan en el intervalo $[1 \cdot 10^{-5}, 1.0]$.

   Además de esto, estos valores se han probado con diferentes criterios aplicados a este factor de aprendizaje. Estos criterios son:

   - **Constante:** El valor del factor de aprendizaje se mantiene constante durante todo el entrenamiento de la red neuronal.
   - **Adaptativo:** El valor del factor de aprendizaje se mantiene constante siempre y cuando el error de la red disminuya a medida que avanza el entrenamiento. En el caso de que este error no disminuya durante dos épocas, el factor de aprendizaje cambiará y será el valor actual dividido entre 5. La idea de esto viene dada a que al no disminuir el error, quiere decir que la red es incapaz de aprender más con ese factor de aprendizaje, por lo que tomar valores más pequeños hará que esta pueda converger mejor en el mínimo de la función del error. No obstante, la reducción de este valor supondrá un aumento en el tiempo de entrenamiento del modelo, ya que al aprender más despacio, será necesario aumentar el número de épocas para poder llegar a los valores de precisión de un modelo con las mismas características con un aprendizaje más rápido (mayor factor de aprendizaje).

   Como se puede apreciar tanto en la Figura 7.14 como en la Tabla 7.15, el tipo de criterio para el factor de aprendizaje no afecta al rendimiento global por lo que es indiferente seleccionar como criterio *constante* o *adaptativo*. No obstante, para el caso del valor del factor de aprendizaje, se puede ver como la precisión global mejora a medida que este aumenta hasta llegar al valor de $1 \cdot 10^{-3}$, donde llega a su valor máximo. A partir de este valor, disminuye considerablemente y se mantiene constante a medida que lo vamos aumentando. Por lo tanto el valor seleccionado para el **factor de aprendizaje es** $1 \cdot 10^{-3}$ con un **criterio de variación *constante***.

3) **Número de épocas:** El número de épocas representa las iteraciones que se van a realizar sobre el conjunto de datos para llevar a cabo el entrenamiento del modelo. Hay que tener cuidado a la hora de determinar un buen número de épocas, ya que si se utiliza un número muy grande es muy probable que el modelo sufra de sobreentrenamiento, lo que supondría que funcionaría muy bien para los datos de entrenamiento, pero a la hora de utilizar datos

Factor de Aprendizaje	Tipo de modificación	Precisión
$1 \cdot 10^{-5}$	**Constante**	0,88983
	Adaptativa	0,88983
$1 \cdot 10^{-4}$	**Constante**	0,91474
	Adaptativa	0,91474
$1 \cdot 10^{-3}$	**Constante**	0,93281
	Adaptativa	0,93281
0,01	**Constante**	0,76003
	Adaptativa	0,76003
0,1	**Constante**	0,76003
	Adaptativa	0,76003
1,0	**Constante**	0,76003
	Adaptativa	0,76003

Tabla 7.15: Precisión Redes Neuronales en función del factor de aprendizaje y su modificación.

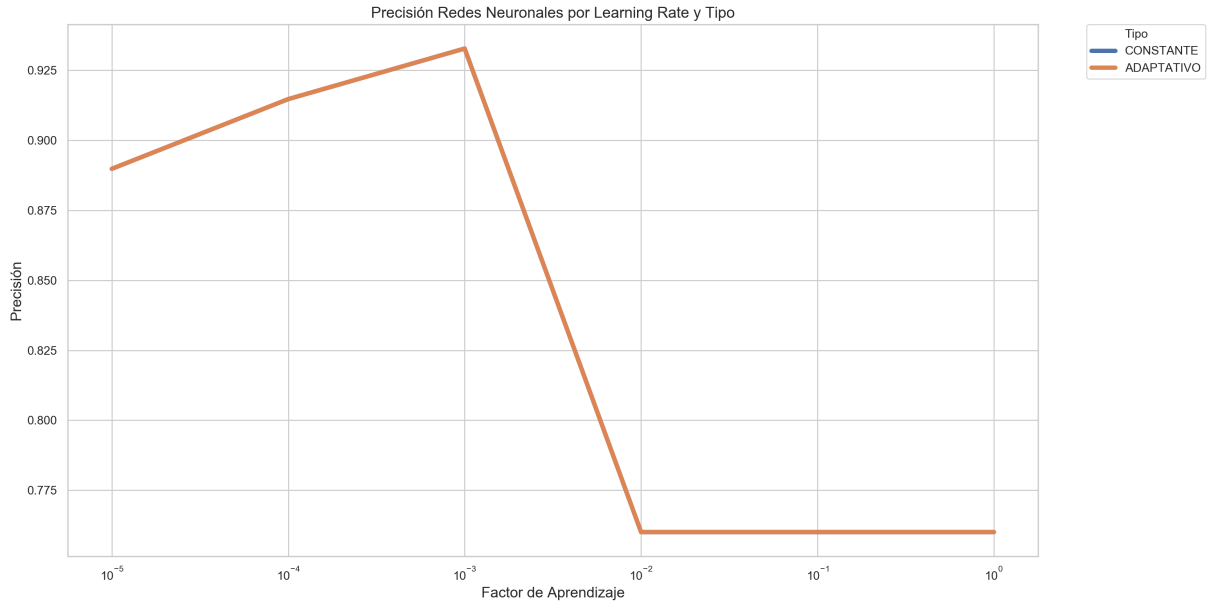


Figura 7.14: Precisión Redes Neuronales en función del factor de aprendizaje.

nuevos, no los clasificaría bien (no es capaz de *generalizar*). Un criterio para determinar cuándo se ha de dejar de aumentar este número es observar la precisión que se obtiene tanto para el entrenamiento como para la validación, de tal forma que en el momento en el que la precisión sobre el conjunto de entrenamiento siga aumentando pero para el conjunto de test comience a disminuir, es el indicativo de que a partir de ese número de épocas la red comienza a sobreentrenar. Esta característica se ve claramente reflejada en la Figura 7.15, en la que se ha entrenado la red con los valores seleccionados hasta ahora con un número distinto de épocas. Para este caso, a partir de las 150 épocas, la red comienza a sobreentrenar por lo cual un valor correcto de épocas consistiría en tomar un valor anterior a este, de tal forma que este no se llegue a aproximar al umbral en el que la red sobreajusta. Por tanto, el valor de

épocas que se va a utilizar es: **número de épocas = 125**.

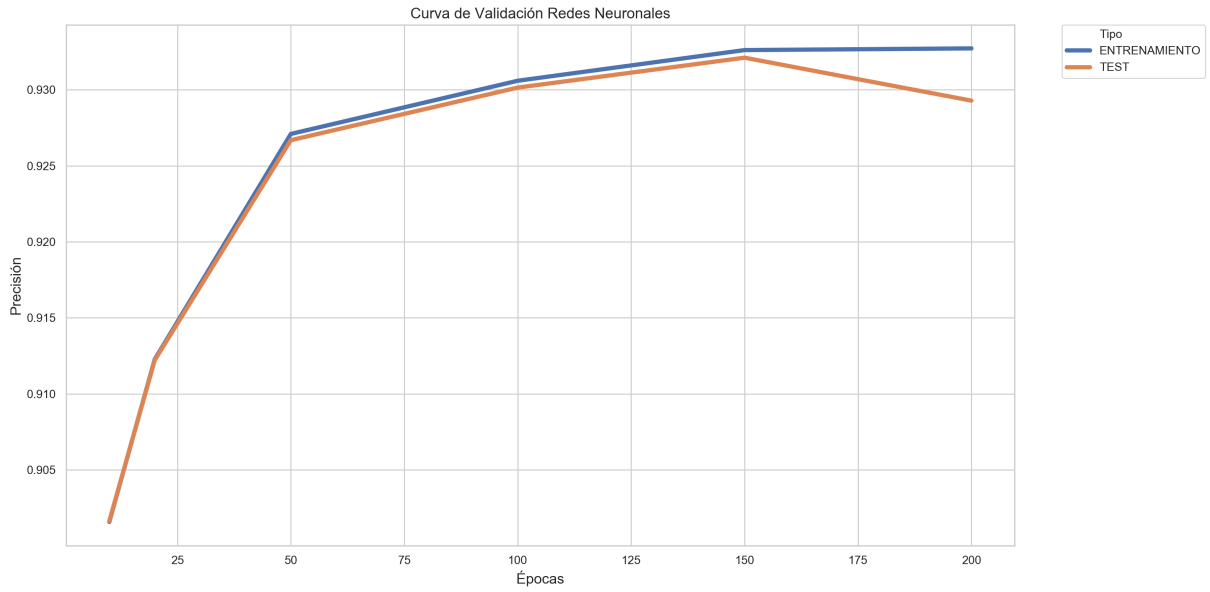


Figura 7.15: Curva de Validación Red Neuronal.

De acuerdo a los resultados obtenidos a lo largo del análisis de los hiperparámetros que conforman las redes neuronales, la configuración final de este es la siguiente (ver Tabla 7.16):

Hiperparámetro	Valor Antiguo	Valor Nuevo
Configuración de la red neuronal	(100)	(38, 25, 40, 35)
Número de épocas	100	125
Factor de Aprendizaje	$1 \cdot 10^{-3}$	$1 \cdot 10^{-3}$
Tipo	Constante	Constante

Tabla 7.16: Modificaciones realizadas en los hiperparámetros de las redes neuronales.

7.2.5. Comparativa

Una vez finalizado el proceso de refinamiento de cada uno de los modelos, es el momento de comparar el rendimiento final tras haber incluido las modificaciones que se han descrito a lo largo de la sección. En la Tabla 7.17, se realiza la comparativa de la precisión obtenida por el modelo inicial (sin modificaciones) y el modelo refinado.

Como se puede observar, en mayor o menor medida, se ha conseguido mejorar la precisión de todos los modelos, siendo el de K-Vecinos Más Cercanos y el de Redes Neuronales los que mayor mejora presentan (3 % y 2 % respectivamente). Por otra parte, también se ha conseguido mejorar el modelo basado en Bosques Aleatorios que, a pesar de que este incremento en la precisión no es muy grande, es muy difícil de llevar a cabo en los rangos de precisión en los que se encontraba (prácticamente del 100 %). Del mismo modo, el modelo de Regresión Logística también ha sufrido mejoras con respecto a su versión base (2 %) pero aún así siguen siendo resultados demasiado bajos en comparación al resto de modelos, por lo que será descartado.

Modelo	Precisión base	Precisión refinada	Mejora
Bosques Aleatorios	0,99668	0,99890	+ 0,00222
K-Vecinos Más Cercanos	0,92401	0,95282	+ 0,02880
Regresión Logística	0,79583	0,81486	+ 0,01903
Redes Neuronales	0,91570	0,93561	+ 0,01990

Tabla 7.17: Comparativa modelo base vs modelo con modificaciones.

Finalmente, y a la vista de los resultados tan altos y balanceados por parte de todos los modelos tras haber sido optimizados, se ha optado por crear un nuevo modelo de *ensemble* (Ver Seccion 5.3.4), el cual se encuentre constituido por estos modelos, consiguiendo así combinar los puntos fuertes de todos ellos y consolidar los resultados obtenidos a partir de los mismos.

Capítulo 8

Evaluación y Resultados

Una vez realizado el proceso de refinamiento de los modelos, es el momento de comprobar qué tal se comportan con nuevos ejemplos, es decir, comprobar que tal *generalizan*. Para ello se ha utilizado la técnica de la validación cruzada (ver Figura 8.1). Esta técnica consiste en dividir aleatoriamente el conjunto de datos de entrenamiento en k subconjuntos distintos e iguales, de tal forma que, en cada una de i iteraciones ($i = 1 \dots k$), se utilizará el subconjunto i como conjunto de prueba, mientras que los otros *k-1* subconjuntos restantes se usarán como conjunto de entrenamiento. De esta forma se consigue que todos los subconjuntos que se generan se utilicen exactamente 1 vez como conjunto de prueba, eliminando el posible sesgo que pudieran tener las muestras y obteniendo el comportamiento del modelo para distintos conjuntos de datos.

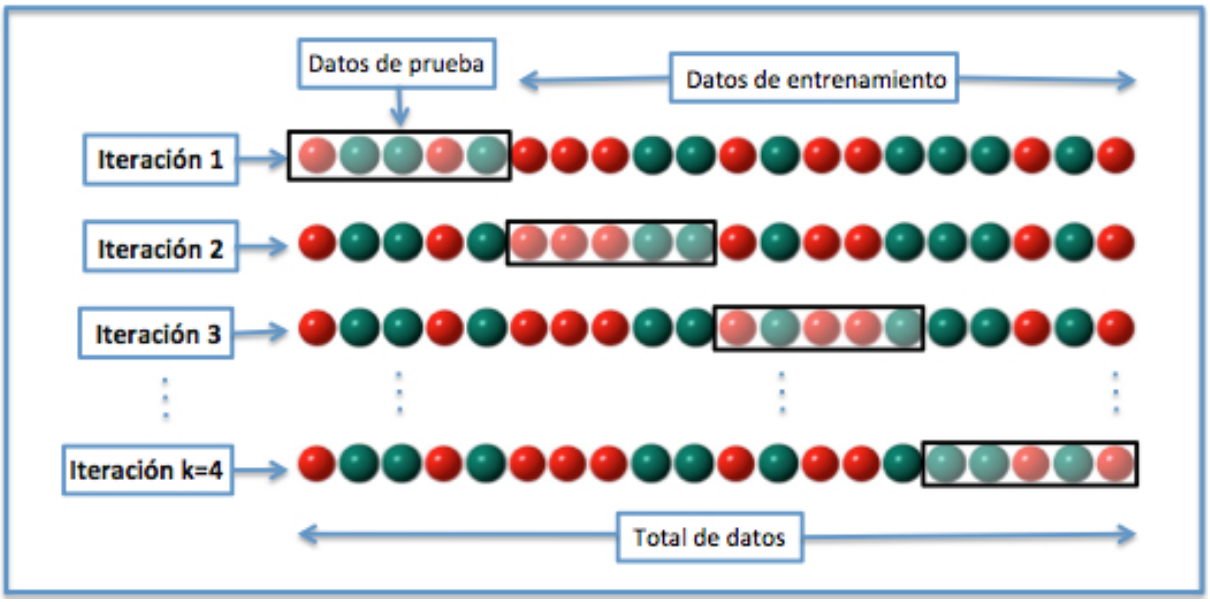


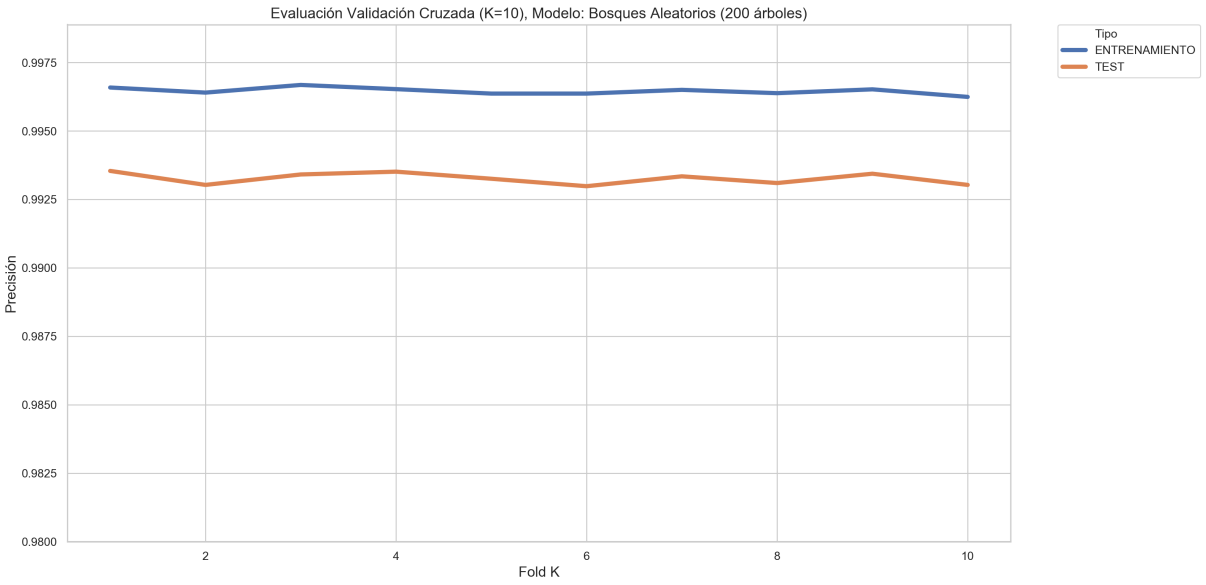
Figura 8.1: Ejemplo Validacion Cruzada.

La aplicación de la técnica de validación cruzada ha sido realizada con un valor: $k = 10$ sobre el conjunto de entrenamiento y realizando una división estratificada del conjunto de datos con el siguiente tamaño: 80 % para entrenamiento y 20 % para prueba.

8.1. Evaluación de los modelos

Antes de aplicar la técnica de validación cruzada con el modelo final, vamos a aplicarla a cada uno de los modelos que lo forman, consiguiendo así una vista global de lo bien o mal que se ha realizado el proceso de división de datos, entrenamiento y refinamiento de cada uno de los modelos.

Bosques Aleatorios

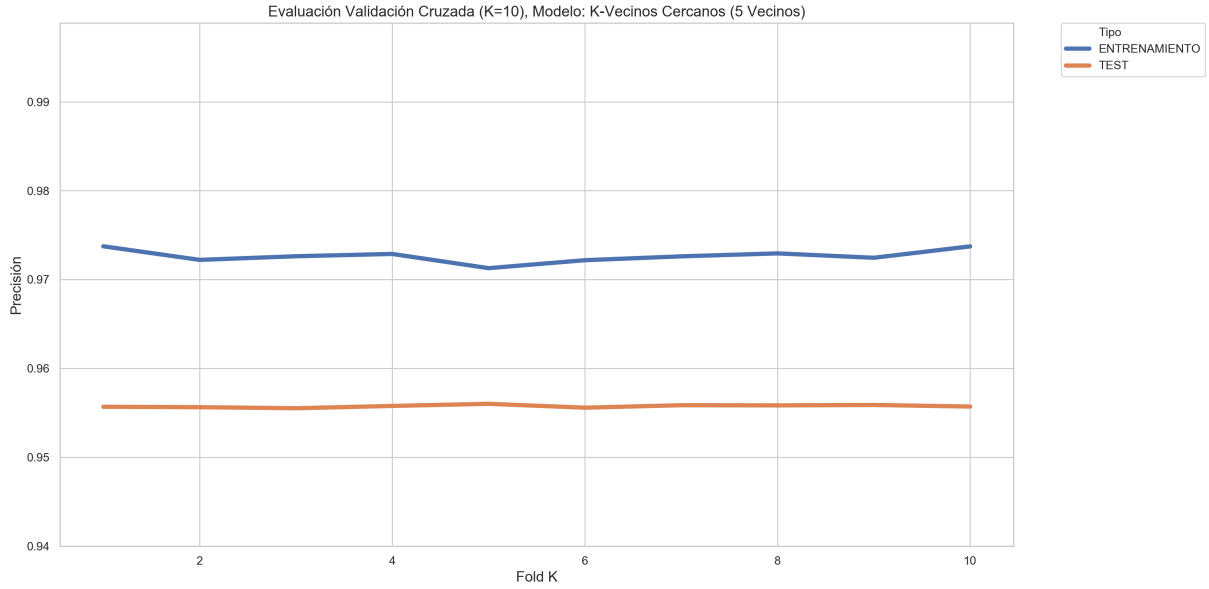


	Media	Desviación estándar
Entrenamiento	0,99645	± 0,00012
Test	0,99126	± 0,00021

Figura 8.2: Evaluación por el método de validación cruzada del modelo: **Bosques Aleatorios**.

Para el modelo basado en bosques aleatorios (ver Figura 8.2), tenemos que la precisión se mantiene prácticamente constante, tanto para el conjunto de entrenamiento como para el conjunto de prueba, además de no existir grandes diferencias entre ellos. Este hecho es una prueba muy clara de que el modelo es capaz de generalizar correctamente el problema ya que funciona prácticamente igual con datos que ya "ha visto" (datos de entrenamiento) como con datos totalmente "nuevos" (datos de prueba). Finalmente, los valores de desviación estándar obtenidos para la media del entrenamiento y la media de test son muy bajos, lo cual significa que los valores de cada una de las k iteraciones varían muy poco con respecto al valor de la media.

K-Vecinos Más Cercanos



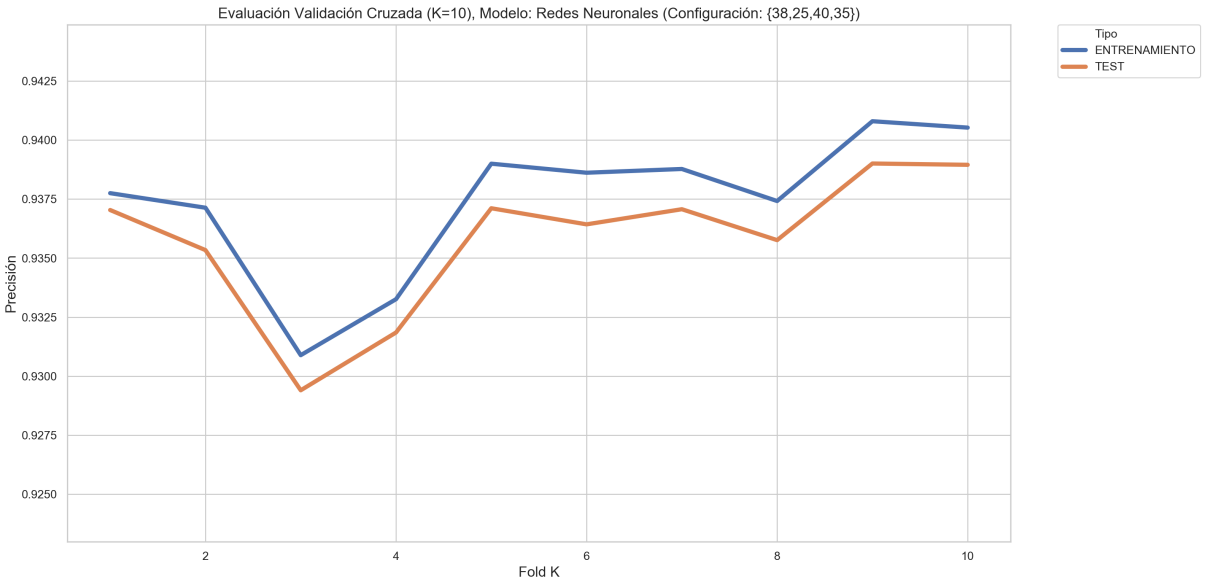
	Media	Desviación estándar
Entrenamiento	0,97741	± 0,00294
Test	0,95579	± 0,00286

Figura 8.3: Evaluación por el método de validación cruzada del modelo: **K-Vecinos Más Cercanos**.

En el caso del modelo del K-Vecinos Más Cercanos, tenemos una situación similar al de Bosques Aleatorios, y es que, la precisión obtenida tanto en el conjunto de entrenamiento como en el conjunto de prueba no tiene apenas diferencias y es muy homogéneo, por lo que la generalización por parte del modelo es muy buena. Además de esto, la desviación estándar de la media tanto para el conjunto de entrenamiento como para el de test es muy pequeña, lo que significa que apenas existe diferencia entre los valores obtenidos en cada una de las iteraciones durante la validación cruzada y el valor medio obtenido, consolidando de este modo el resultado final proporcionado. No obstante, cabe destacar que, a pesar de tener una muy buena precisión, esta se encuentra por debajo de la del resto de modelos probados para esta tarea.

Redes Neuronales

Para el caso del modelo de redes neuronales, podemos observar que la generalización por parte de este es muy buena, debido a que la diferencia de precisión entre los dos conjuntos de datos es muy pequeña y sin apenas variaciones. Además de esto, las desviaciones estándar con respecto a la media tanto para el conjunto de entrenamiento como para el conjunto de test son también muy pequeñas, confirmando la conclusión de que el comportamiento del modelo es muy bueno, homogéneo y robusto, además de encontrarse bien balanceado.



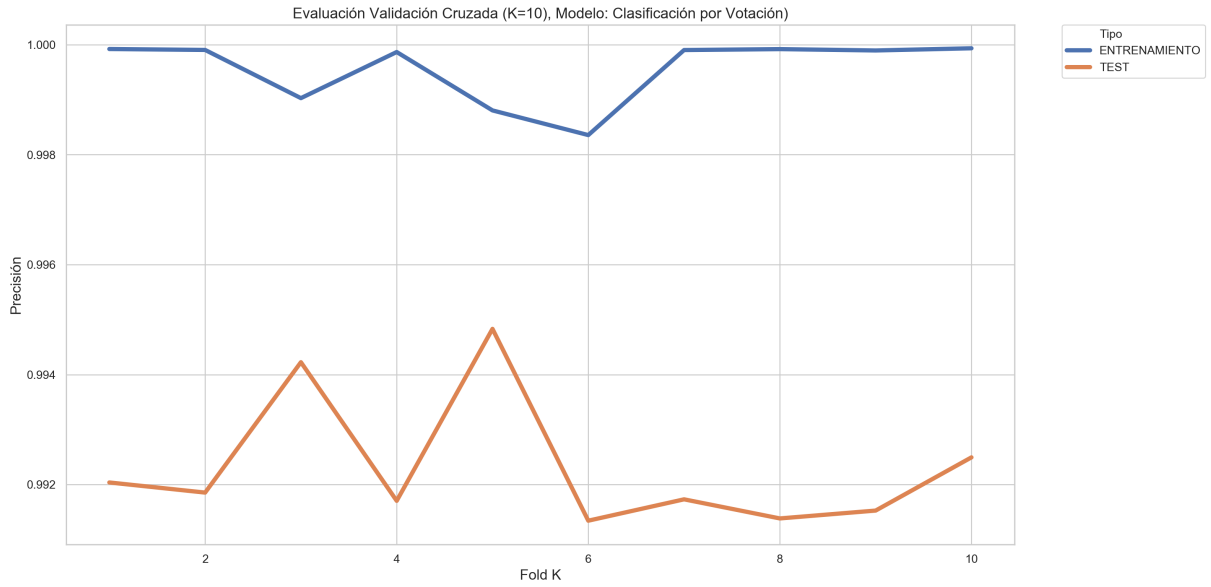
	Media	Desviación estándar
Entrenamiento	0,94265	± 0,00069
Test	0,93573	± 0,00014

Figura 8.4: Evaluación por el método de validación cruzada del modelo: **Redes Neuronales**.

Clasificación por Votación

Finalmente, para el caso del modelo de clasificación por votación, el cual se encuentra constituido por la combinación de los modelos anteriores, vemos cómo los resultados finales son muy similares (e incluso un poco mejores), a los del modelo basado en bosques aleatorios (el cual era el que mejores resultados arrojaba de todos los anteriores). Esto se debe a que al ser un modelo constituido por todos los anteriores y al estar todos ellos bien balanceados, se beneficia de las características de los mismos, por lo que es lógico que el resultado final que se obtiene sea mejor que el de los modelos por separado. Además de los buenos resultados medios obtenidos tanto para el conjunto de entrenamiento como para el de test, hay que añadir su baja desviación estándar en ambos, afianzando así la consistencia y robustez del modelo.

La Figura 8.6 muestra una comparativa global de los resultados obtenidos con la técnica de validación cruzada, aplicada sobre los distintos modelos de aprendizaje automático a lo largo del presente capítulo.



	Media	Desviación estándar
Entrenamiento	0,99955	± 0,00056
Test	0,99231	± 0,00115

Figura 8.5: Evaluación por el método de validación cruzada del modelo: **Clasificación por Votación**.

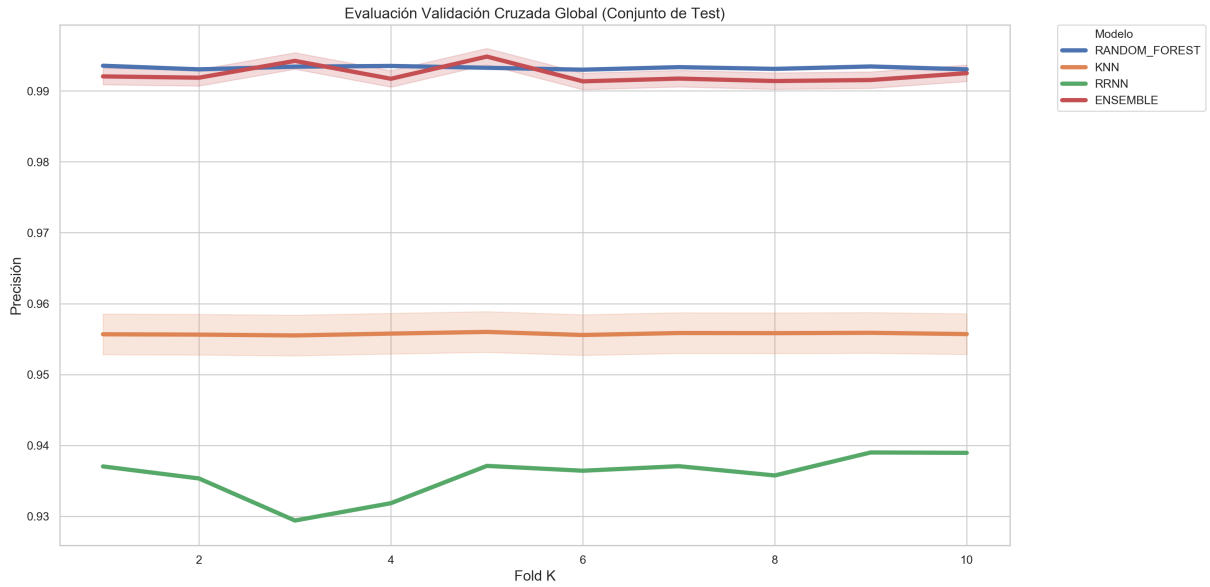


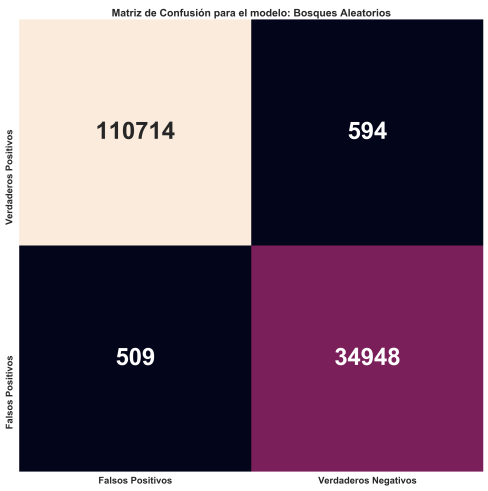
Figura 8.6: Comparativa global del método de validación cruzada para los modelos anteriores.

8.2. Resultados

Por último, tras haber comprobado que todos los modelos funcionan correctamente y son capaces de generalizar el problema tras el entrenamiento, el último paso para acabar de evaluar el desempeño de estos consiste en calcular y analizar las principales métricas (ver Sección 5.3) de evaluación para cada uno de estos.

Para ello, se tomará el subconjunto de prueba (el cual se encuentra constituido por 146.765 registros los cuales han sido seleccionado de manera estratíficada del conjunto total de los datos), y aplicaremos el modelo sobre estos datos, clasificando cada una de las salidas en las 4 clases que conforman la matriz de confusión: *Verdadero Positivo, Verdadero Negativo, Falso Positivo y Falso Negativo*. A partir de estas clases, obtendremos las distintas métricas de los modelos *(precision, recall y f1)*.

Bosques Aleatorios



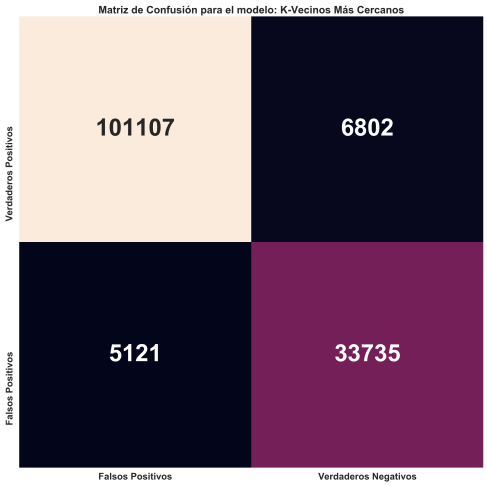
Metrica	Valor
Precision	99,54236 %
Recall	99,46634 %
F1	99,50433 %

Figura 8.7: Matriz de Confusión y Métricas del modelo: **Bosques Aleatorios**.

Para el caso de los bosques aleatorios (ver Figura 8.7), tenemos que los resultados obtenidos se encuentran muy bien balanceados (apenas existe diferencia entre sus métricas), por lo que su eficacia tanto a la hora de detectar como de clasificar las configuraciones es muy buena (99,5 % de media global) y ademas de esto, tiene un *recall* altísimo, lo cual hace que apenas arroje falsos positivos en la clasificación.

K-Vecinos Más Cercanos

En el caso del modelo K-Vecinos Más Cercanos (ver Figura 8.8), tenemos que los resultados finales no son tan homogéneos como los encontrados en otros modelos. Para este caso en concreto, el modelo realiza una buena detección de las configuraciones de las pistas (tiene una buena precisión), pero a la hora de llevar a cabo la clasificación arroja un mayor número de falsos positivos en comparación con los que detecta. A pesar de esto, la diferencia entre la precision y

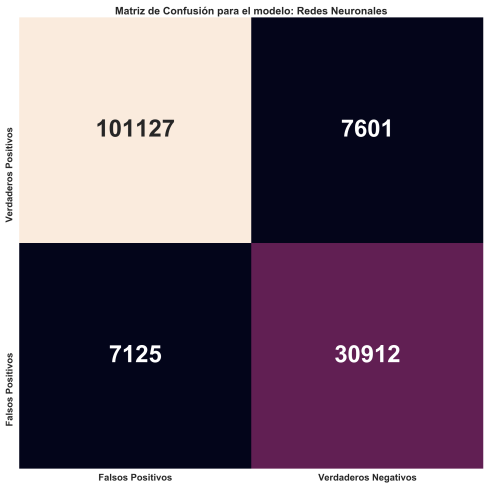


Metrica	Valor
Precision	95,17923 %
Recall	93,69654 %
F1	94,43206 %

Figura 8.8: Matriz de Confusión y Métricas del modelo: **K-Vecinos Más Cercanos**.

el *recall* no es muy elevada por lo que a pesar de que sufra esta peculiaridad, sigue siendo un modelo robusto y con una buena tasa global de clasificación (94,4 %).

Redes Neuronales



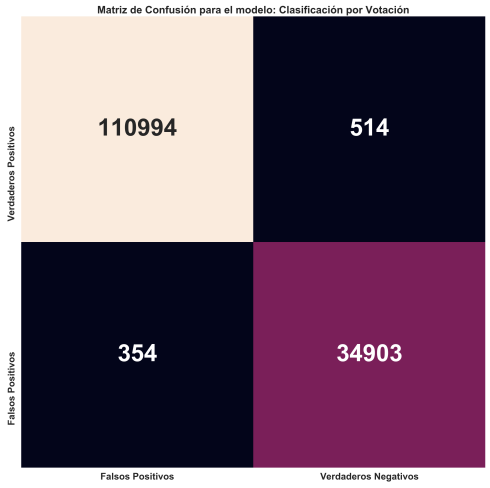
Metrica	Valor
Precision	93,41813 %
Recall	93,00916 %
F1	93,21319 %

Figura 8.9: Matriz de Confusión y Métricas del modelo: **Redes Neuronales**.

Para el modelo basado en redes neuronales (ver Figura 8.9), obtenemos unos resultados finales muy homogéneos ya que no existe apenas diferencia entre las métricas obtenidas. Esto hace que, a pesar de que este modelo tenga una peor eficacia que otros (debido a que sus valores globales son menores), es un modelo que se encuentra bien balanceado, tanto a la hora de detectar la configuración como a la hora de clasificarla.

Clasificación por Votación

Finalmente, para el modelo de clasificación por votación, constituido por todos los modelos anteriores (ver Figura 8.10), vemos como las métricas globales se han conseguido mejorar en todos los aspectos, ya que tanto la precisión como el *recall* toman valores máximos en comparación con otros modelos. Además de esto, los resultados obtenidos son muy uniformes tanto a nivel de precisión como de recall, por lo que su desempeño tanto para la detección como para la clasificación es muy bueno, haciendo de este, la elección de modelo final que se utilizará para resolver esta tarea.



Metrica	Valor
Precision	99,68207 %
Recall	99,53904 %
F1	99,61051 %

Figura 8.10: Matriz de Confusión y Métricas del modelo: **Clasificación por Votación**.

Capítulo 9

Herramienta de Visualización

En el presente capítulo se llevará a cabo el proceso de análisis e implementación que se ha seguido para desarrollar la herramienta de visualización (*dashboard*), así como la funcionalidad de la misma. A pesar de estar realizando un proyecto más orientado a la parte de investigación, el hecho de disponer de una herramienta en la que se pueda tanto interactuar con el modelo de aprendizaje automático, como visualizar información relativa a los vuelos y a las métricas obtenidas a lo largo del desarrollo del proyecto, hace que sea necesaria la labor de realizar una pequeña fase de análisis de la misma.

La fase de análisis comenzará realizando una descripción de los actores implicados, así como la especificación de los distintos tipos de requisitos existentes en un proyecto de desarrollo software (requisitos de usuario, requisitos funcionales, etc). Del mismo modo se realizará la especificación de la arquitectura lógica de la herramienta, así como su diseño e implementación. Finalmente se llevaran a cabo distintas pruebas de caja negra con el objetivo de comprobar que la herramienta es robusta frente a determinas situaciones que pueden ocurrir durante su uso.

9.1. Descripción de los actores

En esta primera sección se abordará la descripción de los distintos actores que interactuarán con nuestra aplicación software. En este caso, al tratarse de un *dashboard* cuya única finalidad consiste en visualizar información de interés para el analista de datos, así como la interacción de este con el modelo de aprendizaje automático, únicamente va a tener un actor, el cual viene descrito en la Tabla 9.1:

A-01	**Usuario**
Versión	1.0 (12/04/2020).
Descripción	Este tipo de usuario se servirá de la aplicación para obtener métricas y datos relevantes para apoyar sus procesos de toma de decisiones, además de interactuar con el modelo de aprendizaje automático.
Comentarios	No necesita autenticación para utilizar la herramienta.

Tabla 9.1: Descripción del Actor-01: Usuario.

9.2. Requisitos de Usuario

En esta sección se presentarán los distintos requisitos de usuario que han de verse satisfechos por la aplicación software, así como las características y peculiaridades de los mismos. En la Tabla 9.2 se muestran los distintos requisitos de usuario asociados al actor *Usuario*:

ID	Nombre
RU-01	Un usuario puede visualizar los datos referentes a los vuelos que se encuentran registrados en el sistema (datos de vuelo y meteorología).
RU-02	Un usuario puede realizar operaciones de filtrado sobre los datos de los vuelos registrados en el sistema.
RU-03	Un usuario puede ver las configuraciones de las pistas que existen en el Aeropuerto de Barajas.
RU-04	Un usuario puede visualizar métricas de evaluación asociadas a los distintos modelos de aprendizaje automático.
RU-05	Un usuario puede interactuar con el modelo para que este realice predicciones en función de los datos que se introduzcan.

Tabla 9.2: Requisitos de Usuario del sistema.

9.3. Casos de Uso

A lo largo de esta sección se presentan los distintos casos de uso asociados a los requisitos de usuario anteriormente descritos, así com la especificación de alguno de ellos y el diagrama de casos de uso ligado al sistema. En la Tabla 9.3 se muestran los distintos casos de uso obtenidos a partir de los requisitos de usuario.

ID	RU Asociado	Nombre
CU-01	RU-01	Listar vuelos.
CU-02	RU-02	Filtrar vuelos.
CU-03	RU-03	Listar configuraciones de pistas.
CU-04	RU-04	Listar métricas de los modelos de aprendizaje automático.
CU-05	RU-05	Predecir configuración de las pistas.

Tabla 9.3: Casos de Uso del sistema.

Una vez obtenidos los casos de uso asociados a los requisitos de usuario, se ha generado el Diagrama de Casos de Uso asociado a cada uno de los actores que interactúan con el sistema, obteniendo una versión esquemática y simplificada de la funcionalidad que ha de satisfacer el sistema. En la Figura 9.1 se muestra el Diagrama de Casos de Uso de la aplicación:

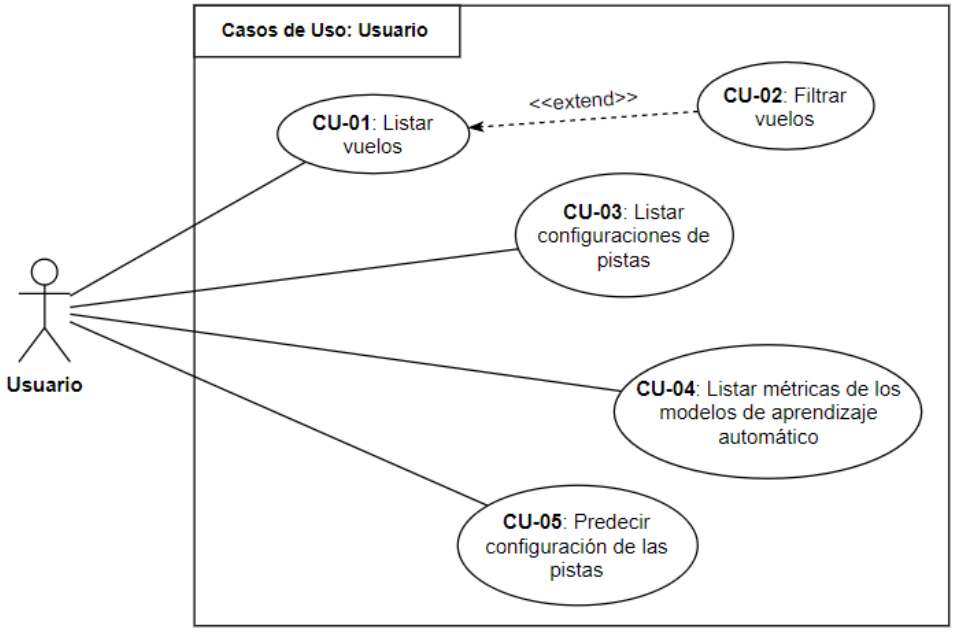


Figura 9.1: Diagrama de Casos de Uso de la aplicación.

9.3.1. Especificación de los casos de uso

En esta subsección se realizará una especificación de los casos de uso obtenidos a partir de los requisitos de usuario. Debido a que la funcionalidad de la mayoría de ellos consiste en realizar un listado de alguna información concreta, únicamente se va a realizar la especificación de uno de ellos (la del resto es totalmente análoga) y la especificación del *CU-05: Realizar predicción de la configuración de las pistas*, ya que además de ser el único caso de uso en el que no se lista información, es el caso de uso en el que se interactúa directamente con el modelo de aprendizaje automático, por lo que su especificación es muy interesante para modelar el flujo a lo largo de esa funcionalidad del sistema.

CU-01	**Listar vuelos**
Versión	1.0 (12/04/2020).
Actor Principal	Usuario.
Actor Secundario	–
Descripción	El sistema permite al usuario visualizar la información relativa a los vuelos que se encuentran registrados en el sistema.
Disparador	El usuario solicita visualizar los vuelos del sistema.
Precondiciones	Los datos de los vuelos han de haberse cargado correctamente al iniciar la aplicación.
Postcondiciones	Ninguna.
Flujo Normal	1. El usuario solicita visualizar los vuelos registrados en la aplicación. 2. El sistema muestra un listado con los datos de los vuelos registrados (ver Vuelo, Operación, Aeronave y Meteorología en la Tabla 6.4). 3. Caso de Uso finalizado satisfactoriamente.
Flujo Alternativo	–
Excepciones	–

Tabla 9.4: Especificación CU-01: Listar vuelos.

CU-05	**Predecir configuración de las pistas**
Versión	1.0 (12/04/2020).
Actor Principal	Usuario.
Actor Secundario	–
Descripción	El sistema permite al usuario realizar una predicción de la configuración de las pistas a partir de los datos introducidos.
Disparador	El usuario solicita llevar a cabo una predicción de la configuración de las pistas.
Precondiciones	–
Postcondiciones	Ninguna.
Flujo Normal	1. El usuario solicita realizar una predicción de la configuración en las pistas. 2. El sistema solicita los datos meteorológicos y del vuelo del que se quiere estimar la configuración de las pistas (ver Vuelo y Meteorología en la Tabla 6.4). 3. El usuario introduce los datos de la meteorología y del vuelo. 4. El sistema valida los datos, realiza la predicción y la muestra por pantalla. 5. Caso de uso finalizado satisfactoriamente.
Flujo Alternativo	3a) El usuario cancela la acción de predecir configuración. -El caso de uso finaliza.
Excepciones	4. Los datos introducidos por el usuario son incorrectos. - El sistema muestra un mensaje de error y vuelve al paso 2.

Tabla 9.5: Especificación CU-05: Predecir configuración de las pistas.

9.4. Requisitos Funcionales

En esta sección se van a tratar los diferentes requisitos funcionales asociados a los requisitos de usuario mencionados anteriormente. Un requisito funcional sirve para definir una función del sistema bien sea software o de alguno de sus componentes. Por este motivo, los requisitos funcionales es un apartado a considerar ya que nos indican cómo se ha de comportarse el sistema acorde a los requisitos descritos. A continuación se mencionan nuevamente los requisitos de usuario del sistema añadiendo a estos sus requisitos funcionales asociados.

ID	Nombre
RF-01	El sistema habilitará la opción de visualizar los datos de los vuelos almacenados (ver Vuelo, Operación, Aeronave y Meteorología en la Tabla 6.4).
RF-02	El sistema mostrará por pantalla información de los distintos vuelos almacenados.
RF-03	El sistema mostrará una diferenciación entre las operaciones realizadas por cada uno de los vuelos (aterrizaje, despegue).
RF-04	El sistema permitirá ordenar los vuelos por cualquier característica que el usuario solicite (ver Vuelo, Operación, Aeronave y Meteorología en la Tabla 6.4).

Tabla 9.6: Requisitos Funcionales asociados al Caso de Uso CU-01: Listado de vuelos registrados en el sistema.

ID	Nombre
RF-05	El sistema habilitará la opción de filtrar los registros de los vuelos registrados en el sistema.
RF-06	El sistema mostrará los resultados tras aplicar el filtro proporcionado por el usuario.

Tabla 9.7: Requisitos Funcionales asociados al Caso de Uso CU-02: Filtrado de los datos de los vuelos registrados en el sistema.

ID	Nombre
RF-07	El sistema habilitará la opción de visualizar las distintas configuraciones del aeropuerto de Barajas.
RF-08	El sistema mostrará por pantalla las distintas configuraciones del aeropuerto de Barajas.

Tabla 9.8: Requisitos Funcionales asociados al Caso de Uso CU-03: Listado de las configuraciones de pistas del Aeropuerto de Barajas.

ID	Nombre
RF-09	El sistema habilitará la opción de visualizar las distintas métricas de los modelos de aprendizaje automático desarrollados.
RF-10	El sistema mostrará por pantalla las distintas métricas asociadas a los modelos de aprendizaje automático desarrollados.

Tabla 9.9: Requisitos Funcionales asociados al Caso de Uso CU-04: Listado de métricas de evaluación de los distintos modelos de aprendizaje automático.

ID	Nombre
RF-11	El sistema habilitará la opción de realizar predicciones de la configuración de las pistas.
RF-12	El sistema solicitará al usuario los datos meteorológicos y de la aeronave para realizar la predicción (ver Meteorología y Aeronave en la Tabla 6.4).
RF-13	El sistema validará los datos introducidos por el usuario.
RF-14	El sistema mostrará un mensaje de error en el caso de que los datos introducidos no sean correctos.
RF-15	El sistema mostrará la configuración de las pistas predicha.

Tabla 9.10: Requisitos Funcionales asociados al Caso de Uso CU-05: Realizar predicción de la configuración de las pistas.

ID	Nombre
RF-16	El sistema permitirá guardar capturas de pantalla de las imágenes y gráficas generadas.
RF-17	El sistema permitirá ampliar y alejar el contenido de los elementos gráficos.

Tabla 9.11: Requisitos Funcionales asociados a los elementos gráficos del sistema.

9.5. Diseño

9.5.1. Arquitectura Lógica

En la siguiente sección se muestra el diseño lógico que tendrá la aplicación que se ha desarrollado. El despliegue de la herramienta se realizará en un servidor (debido a la confidencialidad de los datos, el despliegue se realizará en la máquina proporcionada por el Departamento de Informática de la Universidad de Valladolid) en el que se encuentra alojado tanto la herramienta software y los registros de los vuelos disponibles en el sistema, así como el modelo de aprendizaje automático sobre el que se realizará la interacción con el usuario. Finalmente el acceso a la herramienta se realizará mediante un navegador web. La Figura 9.2 muestra el diseño lógico de la aplicación:

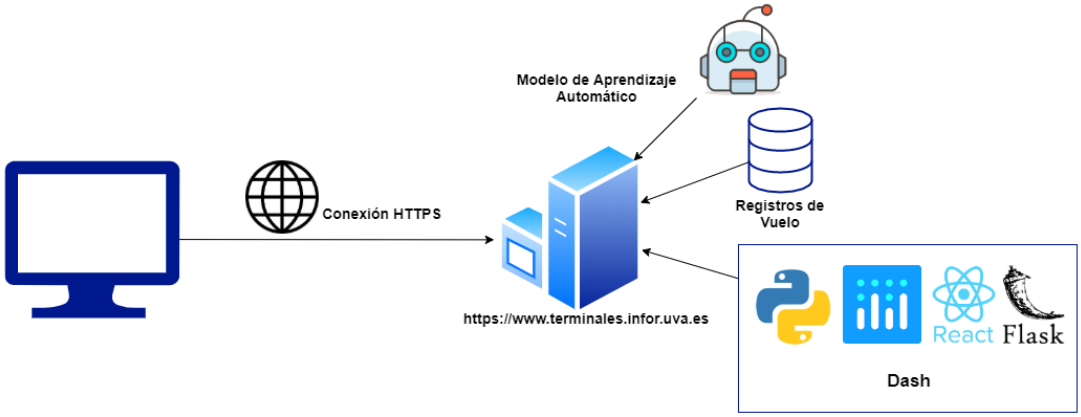


Figura 9.2: Arquitectura Lógica de la aplicación.

9.5.2. Diseño de la aplicación

En esta subsección se muestran las distintas interfaces que conformarán la aplicación final, así como los eventos que se pueden realizar en cada una de ellas:

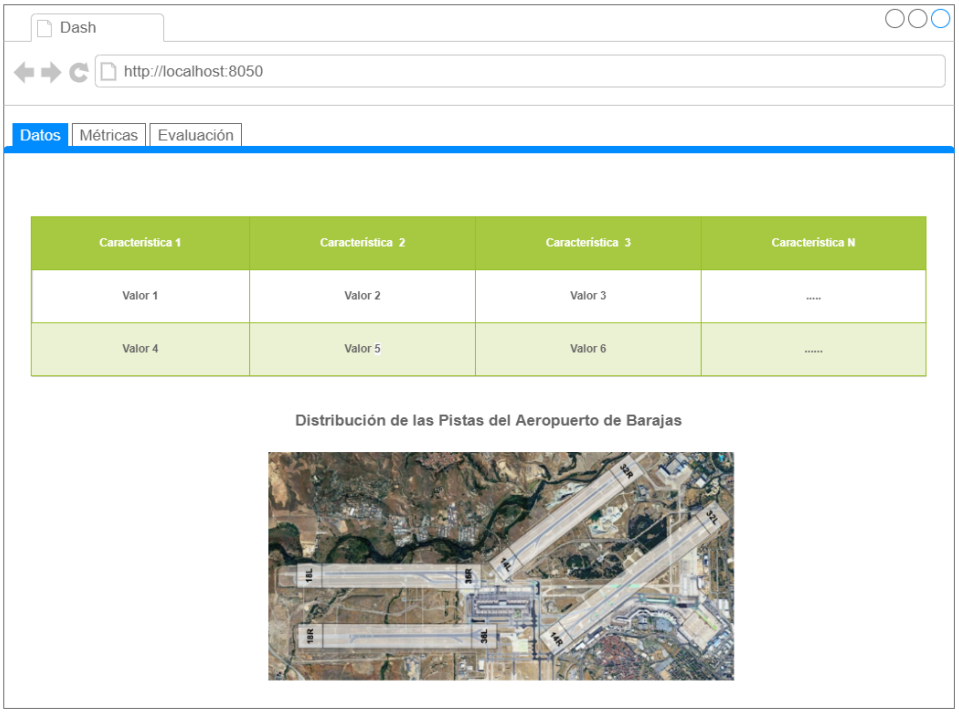
DI-01	**Mostrar datos vuelo**
Descripción	Pestaña encargada de mostrar la información de los vuelos registrados en el sistema y de las configuraciones que dispone el Aeropuerto de Barajas.
Activación	El usuario accede a la pestaña "Datos" situada en la parte superior de la página.
Boceto	
Eventos	Ir a: "Métricas" y a "Evaluación"; Ver Configuración Norte; Ver Configuración Sur.

Tabla 9.12: Diseño Interfaz DI-01: Mostrar datos de vuelo.

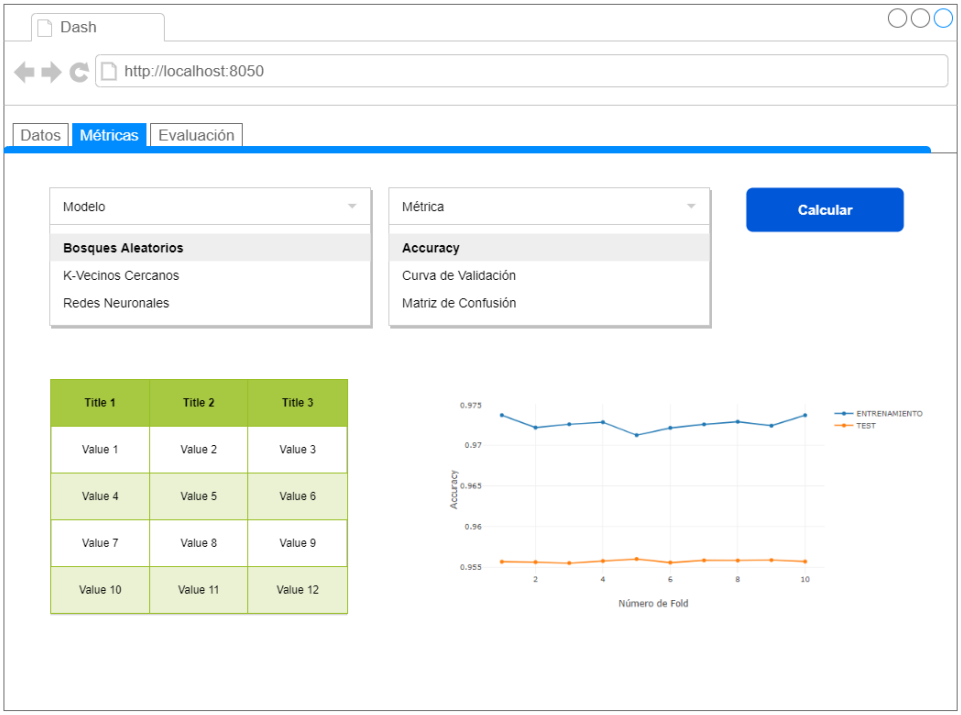
DI-02	Mostrar métricas modelo
Descripción	Pestaña encargada de mostrar al usuario las métricas obtenidas para los distintos modelos de aprendizaje automático.
Activación	El usuario accede a la pestaña "Métricas" situada en la parte superior de la página.
Boceto	
Eventos	Ir a: "Datos" y a "Evaluación"; Visualizar Métricas por Modelo.

Tabla 9.13: Diseño Interfaz DI-02: Mostrar métricas modelos.

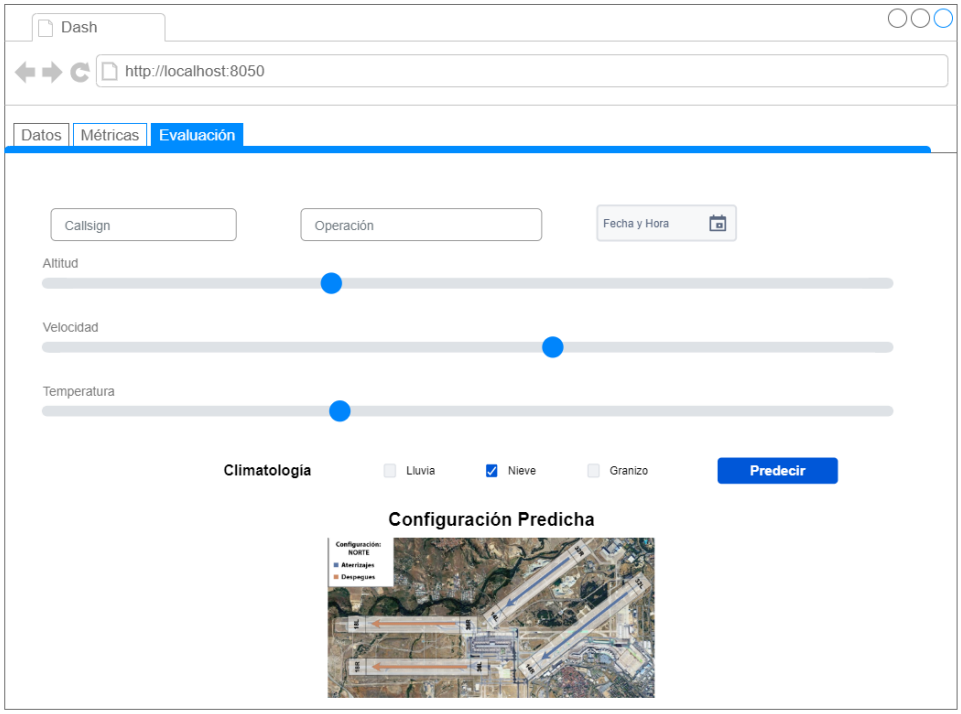
DI-03	Interacción con el modelo de aprendizaje automático
Descripción	Pestaña encargada de mostrar el resultado de la predicción del modelo en función de los datos que se introduzcan en la misma.
Activación	El usuario accede a la pestaña "Evaluación" situada en la parte superior de la página.
Boceto	
Eventos	Ir a: "Datos" y a "Métricas"; Insertar datos vuelo; Realizar predicción.

Tabla 9.14: Diseño Interfaz DI-03: Interacción con el modelo de aprendizaje automático.

9.6. Implementación de la herramienta

En esta sección se definen las principales características y consideraciones que se han tenido en cuenta a la hora de llevar a cabo la fase de desarrollo. Del mismo modo se presenta la funcionalidad y la implementación de las partes más relevantes de la herramienta.

En cuanto a las consideraciones, la principal característica que se ha tenido que seguir a lo largo de todo el desarrollo, ha consistido en trabajar directamente en la máquina virtual, ya que, por motivos de privacidad, los datos relativos a los vuelos debían de mantenerse almacenados en un entorno seguro, privado y gestionado por el Departamento de Informática de la Universidad de Valladolid. Por otra parte, debido a que la herramienta hacía uso de un gran número de elementos gráficos, además de la interacción del usuario con estos, se ha tomado la decisión de emplear **Dash**[1] como framework para desarrollar dicha herramienta. El motivo principal de esta elección reside en que, al tratarse de un framework de Python, la integración tanto con el modelo de aprendizaje automático como con el código desarrollado para el manejo y tratamiento de los datos es del 100 % ya que todos estos componentes comparten un mismo lenguaje de programación: Python.

En cuanto a la implementación, podemos dividir la funcionalidad de la herramienta en 4 clases, en las que, en cada una de ellas, se lleva a cabo una funcionalidad concreta de la herramienta. Estas clases son:

1) **Clase Main:** Es el punto de entrada a la herramienta. Se encarga de crear la instancia del servidor sobre el que se despliega la herramienta. Además, realiza la carga de la vista principal del *dashboard* y de los componentes necesarios para su funcionamiento.

```
flight = FlightData()
metricas = Metricas()
evaluacion = Evaluacion()

external_stylesheets = ["https://codepen.io/chriddyp/pen/dZVMbK.css"]
app = dash.Dash(__name__, external_stylesheets=external_stylesheets)
...
if __name__ == "__main__":
    app.run_server(debug=False)
```

Código 1: Fragmento de la Clase Main

2) **Clase FlightData:** Esta Clase se ocupa de cargar y mostrar los datos relativos a los vuelos que se disponen en el sistema, así como las distintas configuraciones de pistas que dispone el Aeropuerto de Barajas.

3) **Clase Métricas:** Clase encargada de obtener y mostrar los resultados de las distintas métricas asociadas a los modelos de aprendizaje automático desarrollados a lo largo del proyecto.

[1] https://plotly.com/dash/.

```
class FlightData(object):
    def __init__(self):
        self.df_vuelos = pd.read_csv('./data/flights/flightData.csv',
            sep=';')
        self.fight_table = self.generateFightTable()
    def generateFightTable(self):
        return dt.DataTable(
            id='vuelos',
            columns=[{'name':string.capwords(i), 'id':i} for i in
                self.df_vuelos.columns],
            data=self.df_vuelos.to_dict('records'),
            fixed_columns={'headers':True, 'data':5},
            fixed_rows={'headers':True},
            sort_action='native',
            sort_mode='multi',
            filter_action='native',
        )
    ...
```

Código 2: Fragmento de la Clase FlightData

```
class Metricas(object):
    def __init__(self):
        self.metric_df = pd.DataFrame()
    def generate_metric_table(self, model, metric):
        self.metric_df = pd.read_csv(path , sep=';')
        return html.Div(
            children = [
                html.H4("Tabla de Datos"),
                dt.DataTable(
                    id='metric_table',
                    columns=[{'name':string.capwords(i), 'id':i} for i in
                        self.metric_df.columns],
                    data=self.metric_df.to_dict('records'),
                    fixed_columns={'headers':True},
                    fixed_rows={'headers':True},
                    sort_action='native',
                    sort_mode='multi'
            )])
    ...
```

Código 3: Fragmento de la Clase Metricas

4) **Clase Evaluación:** Clase encargada de interactuar con el modelo de aprendizaje automático, de tal forma que, a partir de los datos introducidos, generar la salida con la predicción en la configuración de las pistas.

```
class Evaluacion(object):
    def __init__(self):
        self.data = pd.read_csv('./data/flights/dataCallsign.csv', sep=';')
        # Cargamos el modelo de votacion
        self.final_model = joblib.load('./models/votingClasifier.bin')
    def get_values_from_callsign(self, callsign):
        valores = self.data[self.data['callsign_original'] == callsign]
            .values[0]
        id_leg, hexid, callsign_norm, tipo, origen, destino = valores[0],
            valores[3], valores[4], valores[5], valores[6], valores[7]

        return id_leg, hexid, callsign_norm, tipo, origen, destino
    ...
```

Código 4: Fragmento de la Clase Evaluacion

El código completo del proyecto se puede ver en el contenido adjunto (ver Apéndice A).

9.7. Pruebas de Caja Negra

En esta sección se llevarán a cabo distintas pruebas de caja negra para comprobar que el comportamiento obtenido en puntos críticos de la aplicación es el esperado.

CN-01	Comprobación de la existencia de todos los datos necesarios para predecir la configuración para un nuevo vuelo.
Propósito	Evitar que el modelo falle debido a la falta de datos a la hora de realizar la predicción.
Prerrequisitos	El usuario solicita realizar una predicción de la configuración de las pistas para un nuevo vuelo.
Datos de Entrada	Información meteorológica y datos del vuelo.
Resultado Esperado	Mensaje de error motivado por la falta de alguno de estos campos.
Resultado Obtenido	Mensaje de error debido a que faltan datos para llevar a cabo la predicción.

Tabla 9.15: Prueba de Caja Negra CN-01: Comprobación existencia de todos los datos necesarios para predecir la configuración para un nuevo vuelo.

CN-02	Existencia de los valores de una métrica para un modelo.
Propósito	Evitar que la aplicación falle si el modelo seleccionado no dispone de una determinada métrica.
Prerrequisitos	El usuario solicita visualizar una métrica de un determinado modelo de aprendizaje automático.
Datos de Entrada	Modelo de aprendizaje automático y tipo de métrica.
Resultado Esperado	Mensaje de error debido a la falta de valores para esa métrica.
Resultado Obtenido	Mensaje de error al no disponer de esa métrica para el modelo.

Tabla 9.16: Prueba de Caja Negra CN-02: Existencia de los valores de una métrica para un modelo.

CN-03	Validación del Callsign a la hora de predecir un vuelo.
Propósito	Evitar que el modelo falle debido a una introducción errónea de los datos del vuelo.
Prerrequisitos	El usuario solicita realizar una predicción de la configuración de las pistas.
Datos de Entrada	Callsign.
Resultado Esperado	Mensaje de error debido a la inserción de un Callsign no válido.
Resultado Obtenido	Mensaje de error por un valor de Callsign inválido.

Tabla 9.17: Prueba de Caja Negra CN-03: Validación del Callsign a la hora de predecir un vuelo.

CN-04	Filtrado de los datos de vuelo en función de la configuración asignada.
Propósito	Validar que la herramienta filtra los datos de los vuelos que se llevan a cabo en las pistas con una determinada configuración.
Prerrequisitos	El usuario solicita filtrar vuelos en función de la configuración.
Datos de Entrada	Configuración por la qué se quiere filtrar (Norte, Sur).
Resultado Esperado	Visualización de la información de los vuelos que se han realizado en la configuración proporcionada como entrada.
Resultado Obtenido	Información de los vuelos asignados a la configuración proporcionada como entrada.

Tabla 9.18: Prueba de Caja Negra CN-04: Filtrado de los datos de vuelo en función de la configuración asignada.

Capítulo 10

Conclusiones y Trabajo Futuro

10.1. Conclusiones

A lo largo de esta memoria se ha realizado la documentación técnica del proyecto desarrollado como Trabajo Fin de Grado, en el que se han seleccionado y optimizado diversos algoritmos de aprendizaje automático con el objetivo de brindar al controlador aéreo una herramienta que le ayude a la hora de determinar qué configuración de pistas ha de elegir. Para llevar esto a cabo, se han fijado diversos objetivos cuya consecución nos permitiesen alcanzar esta meta. Es por ello por lo que vamos a realizar un repaso sobre estos para ver si se han conseguido lograr dichos objetivos.

En primer lugar, se realizó un análisis exploratorio de los datos pertenecientes a las transacciones aéreas realizadas en el Aeropuerto de Barajas durante los años 2018 y 2019, en el que se detectaron numerosas incidencias debido a la ausencia de datos, principalmente en factores meteorológicos, los cuales fueron solventados de manera satisfactoria aplicando técnicas de tratamiento de datos ausentes. Además de esto, debido a que los valores de la mayoría de las características eran cualitativos, fue necesario realizar una tarea de transformación y normalización de los mismos para así poder utilizarlos en los distintos modelos de aprendizaje automático. Una vez constituido el modelo de datos final, surgió un gran problema ya que no se disponía de información relativa a la configuración en los registros de los vuelos, lo que supuso generar esta característica de manera artificial, en función de las configuraciones predeterminadas en el Aeropuerto de Barajas y de la utilización de las mismas en periodos de tiempo de 30 minutos. Finalmente, se realizó un análisis de este modelo de datos para observar las posibles relaciones y dependencias existentes en los mismos, además, debido a las particularidades de los modelos de aprendizaje automático, se desarrollaron cuatro conjuntos de datos distintos para probar con los modelos y seleccionar el que mejor resultados arroje.

Tras verificar la consistencia de estos nuevos conjuntos de datos, se realizó una prueba inicial con los principales algoritmos de aprendizaje automático, en la que se comprobó que de todos los modelos de datos utilizados, los que mejores resultados obtenían eran aquellos en los que se utilizaban todas las características de los datos. Esto se tradujo en que, a nivel de datos, se empleara el modelo constituido por todas las características. De manera síncrona, se realizaron pruebas de eficacia para distintos modelos de aprendizaje automático, obteniendo para casi todos ellos unos resultados muy prometedores, por lo que se decidió no descartar ninguno en este punto y ver hasta donde se podrían mejorar. En este punto del flujo y teniendo claro tanto el modelo de

datos a utilizar como los algoritmos, se realizó el refinamiento de los modelos, consiguiendo una mejora de todos estos y descartando aquellos cuyos resultados finales no eran lo suficientemente buenos en comparación al resto.

Por otra parte, se aplicaron distintas técnicas de evaluación sobre los modelos ya optimizados con el fin de ver qué tan bien generalizaban el problema de la selección de la configuración de las pistas, para lo cual se demostró que todos los modelos generalizaban correctamente, por lo que se optó por utilizar como modelo final un meta-clasificador constituido por todos estos modelos, generando un nuevo modelo más robusto y aprovechando las ventajas de los modelos independientes.

Finalmente se implementó un *dashboard* que permitiese tanto la visualización de las métricas de los modelos que se han ido obteniendo a lo largo del proyecto, como la interacción del usuario con el propio modelo. Esta interacción se lleva a cabo mediante un pequeño formulario en el que el usuario introduce las características referentes al vuelo (simulando los datos de entrada) y es el modelo el que arroja la configuración predicha para esa casuística.

Como resumen de estas conclusiones, se han logrado cumplir todos los objetivos marcados al inicio del proyecto, consiguiendo desarrollar un modelo de aprendizaje automático que permita resolver esta problemática que, como se ha podido observar a lo largo del presente documento, es una elección que se encuentra condicionada por una gran cantidad de factores externos, por lo que cabe destacar los buenos resultados que se han obtenido a lo largo de las distintas etapas del desarrollo. Otro resultado llamativo ha sido la diferencia de precisión obtenida con los diferentes conjuntos de datos empleados, siendo más eficaz aquellos en los que se utilizan todas las características de los datos, en lugar de utilizar solamente aquellos que tienen una mayor influencia sobre la configuración de la pista. Finalmente cabe destacar la gran homogeneidad y robustez del modelo final para clasificar nuevos vuelos, siendo esta la principal característica a destacar junto con sus resultados, los cuales son muy prometedores teniendo en cuenta la complejidad del problema a tratar. No obstante sería muy interesante el analizar y comprobar el comportamiento de este modelo en aeropuertos con otras configuraciones, ya que las condiciones en otros aeropuertos serán muy diferentes a aquellas en las que los modelos desarrollados se han mostrado efectivos: diferente número de pistas, mayor variedad de configuraciones, condiciones meteorológicas propias del aeropuerto (mayor lluvia, peor visibilidad, diferentes condiciones de viento, etc), mayor volumen de tráfico aéreo... Probar estos modelos en otros aeropuertos nos permitirá validar que son efectivos independientemente de las condiciones de operación y consolidaría la calidad de nuestros resultados.

10.2. Trabajo Futuro

A pesar de que tras la finalización de este trabajo se ha entregado tanto un modelo como un software totalmente funcional, existen distintas líneas de trabajo futuro que por motivos de tamaño y de alcance no han podido ser llevadas a cabo en este trabajo. Estas líneas de trabajo futuro son:

1) **Aplicación del modelo en otros aeropuertos:** A pesar de que los resultados finales proporcionados por el modelo son muy buenos, una línea muy interesante a profundizar consiste en analizar el comportamiento de este modelo en otros aeropuertos, cuya cantidad y variabilidad de configuraciones sea mayor a la del Aeropuerto de Barajas (el cual únicamente dispone de dos configuraciones distintas).

2) **Utilización de modelos basados en Aprendizaje Profundo:** Por otra parte, otro trabajo futuro muy interesante que debido a su tamaño no pudo incluirse en el alcance de este proyecto, es la utilización de modelos basados en Aprendizaje Profundo para resolver la problemática de la configuración de las pistas. Por lo general, este tipo de modelos suelen obtener unos mejores resultados que los modelos tradicionales, por lo que tratar de encontrar solución a este problema utilizando modelos basados en aprendizaje profundo es una de las líneas de investigación que se quedan abiertas trás la realización de este proyecto.

10.3. Aprendizaje personal

En esta pequeña sección trato sobre la experiencia y aprendizaje personal que ha supuesto para mí la realización de este Trabajo Fin de Grado:

1) He podido mejorar mi habilidad con el lenguaje de programación Python, así como las particularidades de las estructuras de datos y la funcionalidad de las librerías que he utilizado para resolver este problema, en las que destaco Pandas, Numpy y Scikit-learn.

2) Debido a la profundidad en la que se han realizado algunos análisis a lo largo del proyecto, tanto en el conjunto de datos como en los modelos de aprendizaje automático, he aprendido a detectar y extraer información valiosa de los mismos, así como mejorar mi razonamiento y capacidad para elaborar las conclusiones asociadas a cada uno de estos análisis.

3) La realización de este documento en LaTeXme ha permitido mejorar mi habilidad con el uso de este lenguaje, permitiéndome realizar documentación científica con mayor calidad.

4) La realización de este proyecto bajo el marco de trabajo de UVagile me ha aportado experiencia a la hora de gestionar un proyecto utilizando una metodología ágil, así como mejorar mi capacidad de autogestión y organización, permitiéndome tener un mayor control tanto sobre el avance del mismo como de las tareas que quedaban pendientes.

5) He aprendido cómo realizar completamente la creación de un modelo de aprendizaje automático y optimizarlo, ya que a pesar de que el flujo de trabajo es relativamente sencillo, existen pequeñas características a lo largo del mismo que hacen que el resultado final sea muy distinto en el caso de que se realice de la forma incorrecta.

6) He mejorado mis conocimientos y habilidades a la hora de realizar *dashboards* interactivos utilizando la librería Dash ya que, a pesar de que era una librería de la que tenía constancia, nunca había tenido la posibilidad de trabajar con ella de primera mano.

Bibliografía

[1] Federal Aviation Administration. What is NextGen?, (visitado 13-02-2020). https://www.faa.gov/nextgen/what_is_nextgen/.

[2] Airport Technology. Single European Sky: can digitalisation improve air traffic in Europe?, (visitado 11-02-2020). https://www.airport-technology.com/features/single-european-sky-latest-developments/.

[3] Álvaro Alonso-Isla, Pedro C. Álvarez-Esteban, Aníbal Bregón, Luís D'Alto, Fernando Díaz, Iván García, Paula Gordaliza, Javier López-Leonés, Miguel A. Martínez-Prieto, David Scarlatti, and Miguel Vilaplana. AIRPORTS: Análisis de eficiencia operacional basado en trayectorias de vuelo. In *Actas de las XXIII Jornadas de Ingenieria del Software y Bases de Datos, JISBD*, 2018.

[4] Jacob Avery and Hamsa Balakrishnan. Predicting airport runway configuration: A discrete-choice modeling approach. *Proceedings of the 11th USA/Europe Air Traffic Management Research and Development Seminar, ATM 2015*, 2015.

[5] Jacob Avery and Hamsa Balakrishnan. Data-driven modeling and prediction of the process for selecting runway configurations. *Transportation Research Record*, 2600(2600):1–11, 2016.

[6] Dimitris Bertsimas, Michael Frankovich, and Amedeo Odoni. Optimal selection of airport runway configurations. *Operations Research*, 59(6):1407–1419, 2011.

[7] Ministerio de Fomento de España. Consolidación reglamento de circulación aérea, (visitado 15-03-2020). http://www.fomento.gob.es/NR/rdonlyres/8E98FA39-28AD-46FD-AAAE-68114B0FE5D4/128215/Txt_Consldd_RCA20141218.pdf.

[8] Arjen de Leege and Ceriel Janssen. Probabilistic runway and capacity forecasting using machine learning to support decision making. *SESAR Innovation Days*, (November), 2016.

[9] Dakshina Dhanasekaran. Improved Prediction of Runway Usage for Noise Forecast. 2014.

[10] Iván García, Miguel A. Martínez-Prieto, Anibal Bregón, Pedro C. Álvarez, and Fernando Díaz. Towards a Scalable Architecture for Flight Data Management. In *Proceedings of the 6th International Conference on Data Science, Technology and Applications*, pages 263–268, 2017.

[11] Air Transport Action Group. Key facts and figures from the world of air transport, (visitado 11-03-2020). https://aviationbenefits.org/media/166344/abbb18_full-report_web.pdf.

[12] Yael Grushka-Cockayne and Bert De Reyck. Towards a single european sky. *Interfaces*, 39(5):400–414, 2009.

[13] Robert Horonjeff, Francis McKelvey, William Sproule, and Seth Young. *Planning and Design of Airports*. 2010.

[14] Gonzalo-Nicolás Marenco Alemán. Funcionamiento y Evolución de Aeropuertos ante una Demanda Turística creciente. 2010.

[15] Miguel A. Martínez-Prieto, Anibal Bregon, Iván García-Miranda, Pedro C. Álvarez-Esteban, Fernando Díaz, and David Scarlatti. Integrating flight-related information into a (Big) data lake. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2017-September, 2017.

[16] Miguel A Martinez Prieto, Jorge Silvestre Vilches, Aníbal Bregón Bregón, and José Ignacio Farrán Martín. Hacia la Consolidación de las Aulas Ágiles. *Actas de las XXVI Jornadas sobre la Enseñanza Universitaria de la Informática (JENUI)*, pages 29–36, 2020.

[17] Earth Networks. The ultimate guide to help airports, airlines, and fixed-base operators (FBOs) navigate lightning's many safety and operational challenges., (visitado 21-02-2020). https://www.earthnetworks.com/the-ultimate-lightning-guide-for-airport-operations/.

[18] Joan Nusbaum. Cold Weather Compost.

[19] Continuous Descent Operations. Executive summary Probabilistic 2-Day Forecast of Runway Use. (June):14–17, 2011.

[20] Organización de Aviación Civil Internacional. *Anexo 14 al Convenio sobre Aviación Civil Internacional. Aeródromos*. 2009.

[21] Nicolas Peña and Pedro C Álvarez Ésteban. An Algorithm to Determine Airport Runway Usage / Configuration Based on Aircraft Trajectories. *38th Digital Avionics Systems Conference (DASC)*, 2019.

[22] Javier R. Ventosa. ENAIRE se prepara para el futuro. *Revista del Ministerio de Fomento*, pages 12–15, 2017.

[23] Varun Ramanujam and Hamsa Balakrishnan. Estimation of maximum-likelihood discrete-choice models of the runway configuration selection process. *Proceedings of the American Control Conference*, pages 2160–2167, 2011.

[24] Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning: Aprendizaje automático y aprendizaje profundo con Python, scikit-learn y TensorFlow*. 1 edition, 2019.

[25] Miquel Rodríguez. Estimando con Planning Poker y puntos de historia, Story Points, (visitado 26-02-2020). https://www.netmind.es/knowledge-center/estimando-con-planning-poker-y-story-points.

[26] Joaquín Amat Rodrigo. Regresión logística simple y múltiple, (visitado 20-04-2020). https://www.cienciadedatos.net/documentos/27_regresion_logistica_simple_y_multiple.

[27] Janet Saavedra Vera. *Puertos Y Aeropuertos*. 2013.

[28] Ken Schwaber and Jeff Sutherland. The Scrum Guide, (visitado 19/04/2020). https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf.

[29] SkyBray. Runway End Safety Area, (visitado 15-04-2020). https://www.skybrary.aero/index.php/Runway_End_Safety_Area.

[30] Martin Strohmeier, Matthias Schäfer, Vincent Lenders, and Ivan Martinovic. Realities and challenges of nextgen air traffic management: The case of ADS-B. *IEEE Communications Magazine*, 52(5):111–118, 2014.

[31] Axel Tanner and Martin Strohmeier. Anomalies in the Sky: Experiments with traffic densities and airport runway use. *Proceedings of the 7th OpenSky Workshop*, pages 51–62, 2019.

[32] ADS-B Technologies. ADS-B Systems, (visitado 08-04-2020). http://www.ads-b.com/.

[33] Trever. IFR vs VFR: What's the Difference between these Two Flying Methods?, (visitado 19-02-2020). https://inflightpilottraining.com/2019/04/19/ifr-vs-vfr-whats-the-difference-between-these-two-flying-methods/.

Apéndices

Apéndice A

Contenido Adjunto

En esta sección del apéndice se detalla el contenido anexo junto a este documento en la entrega del Trabajo Fin de Grado. En este contenido adjunto se incluyen los scripts necesarios para realizar todas las operaciones que se han descrito a lo largo del presente documento: carga y normalización de datos, obtención de nuevas características a partir de estos, prueba y optimización de los modelos de aprendizaje automático, evaluación de los resultados, etc. Además de esto, se incluye también el fichero binario del modelo de aprendizaje automático final, así como la herramienta software que permite la interacción con este.

El contenido adjunto sigue la siguiente estructura de directorios:

- /**app:** En este directorio se encuentra alojado el código necesario para ejecutar la aplicación software (ver Figura A.1). Se encuentra constituido por los siguientes subdirectorios:
  - /**data:** En este subdirectorio se encuentran los ficheros de datos necesarios para visualizar los registros de los vuelos, obtener las métricas de los distintos modelos de aprendizaje automático, etc.
  - /**img:** En este subdirectorio se alojan las distintas imágenes que se utilizarán en la herramienta software.
  - /**modules:** Subdirectorio en el que se encuentra alojado el código modularizado para llevar a cabo las diferentes funcionalidades de las que dispone la herramienta.
  - **app.py:** Fichero de entrada (main) a la aplicación software.
- /**memoria:** Directorio en el que se encuentra la memoria del Trabajo Fin de Grado.
- /**scripts:** En este directorio se encuentra alojado el código necesario para llevar a cabo cada una de las operaciones descritas a lo largo de la memoria. A su vez, este se encuentra estructurado en distintos subdirectorios, en los que se almacena el código en función de la tarea que realiza (ver Figura A.2). Estos subdirectorios son:
  - /**configuracionPistas:** Subdirectorio en el que se encuentra el código necesario para llevar a cabo la obtención de la configuración de las pistas a partir de los datos originales.
  - /**correlacionDatos:** En este subdirectorio se encuentra el código para generar las matrices de correlación y estudiar la relación entre las características de los datos.

- **/curvaValidacion:** Subdirectorio en el que se encuentran el código y los resultados obtenidos trás realizar el análisis mediante curvas de validación a los distintos modelos de aprendizaje automático.
- **/data:** Subdirectorio en el que se encuentran los datos de los resultados obtenidos en las distintas fases llevadas a cabo a lo largo del proyecto. Por motivos de confidencialidad no se incluirá ningún dato relativo a los registros de vuelo.
- **/entrenamientoModelos:** En este subdirectorio se encuentra el código necesario para generar y entrenar los distintos modelos de aprendizaje automático en su configuración final, así como alojar los binarios generados tras el entrenamiento de los mismos.
- **/limpiezaDatos:** Subdirectorio en el que se encuentra el código para llevar a cabo las operaciones de limpieza, preprocesamiento y normalización de los datos.
- **/matrizConfusion:** En este subdirectorio se encuentra el código necesario para evaluar los distintos modelos utilizando la matriz de confusión, así como los resultados obtenidos en la misma.
- **/optimizacionModelos:** Subdirectorio en el que se encuentra el código para obtener el resultado inicial de los distintos modelos de aprendizaje automático, así como la optimización y refinamiento de los mismos.
- **/validacionCruzada:** Subdirectorio en el que se encuentran el código y los resultados obtenidos trás realizar el análisis mediante validación cruzada a los distintos modelos de aprendizaje automático.

- **/requirements.txt:** Fichero en el que se encuentran las dependencias necesarias para ejecutar el proyecto.

```
/
└── app/
    ├── data/ --> Datos que utiliza la aplicación
    │   ├── flights/ --> Datos de los vuelos
    │   │   ├── dataCallsign.csv
    │   │   └── flightData.csv
    │   ├── metric/ --> Datos de los modelos
    │   │   ├── ensemble_CV.csv
    │   │   ├── KNN_Accuracy.csv
    │   │   └── ⋮
    │   └── model/ --> Binario del modelo
    │       └── votingClasifier.bin
    ├── img/ --> Imágenes que utiliza la aplicación
    ├── modules/ --> Componentes para la funcionalidad de la herramienta
    │   ├── DashUtils.py
    │   ├── Evaluacion.py
    │   ├── FlightData.py
    │   └── Metricas.py
    └── app.py --> Punto de entrada a la herramienta
```

Figura A.1: Estructura de subdirectorios del directorio: */app*.

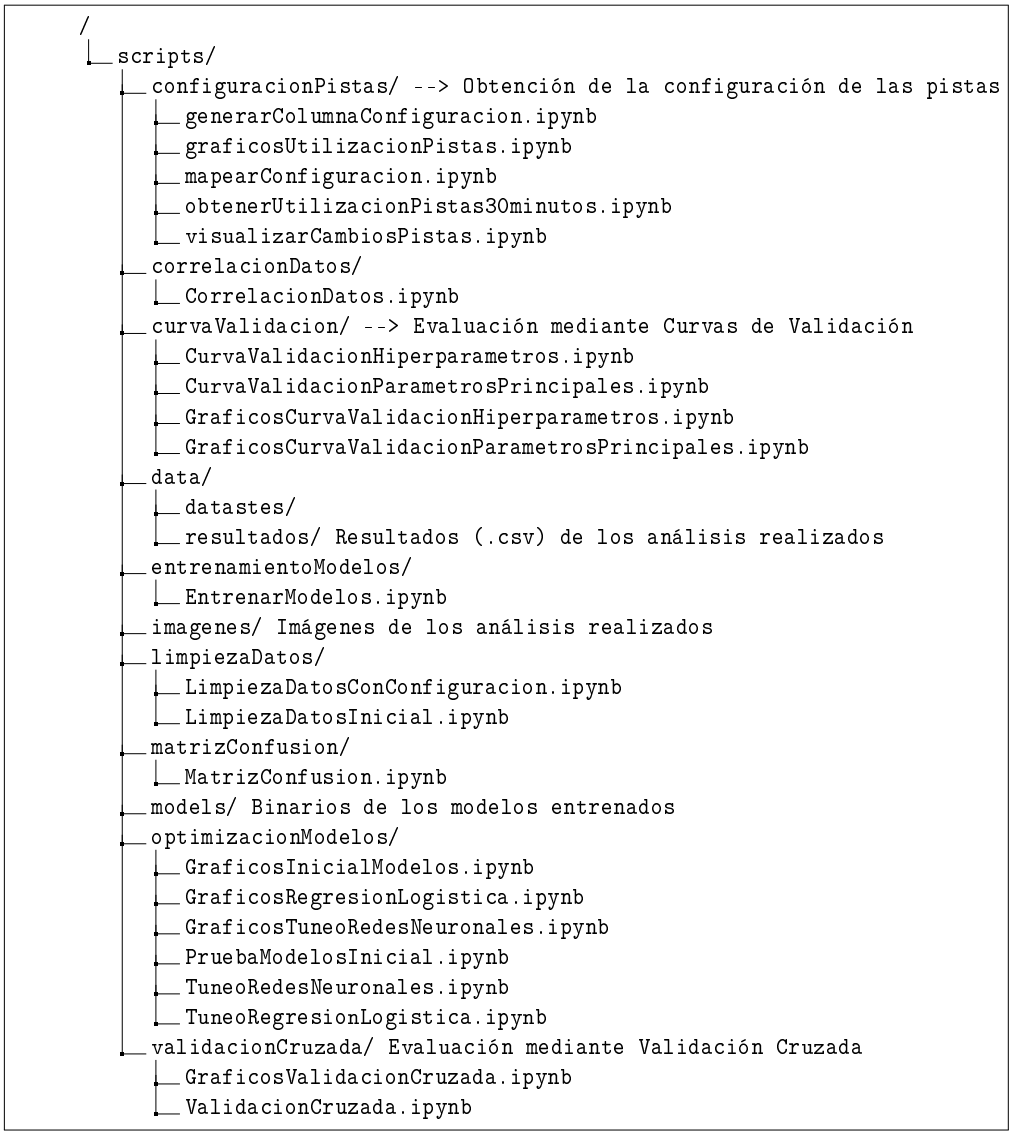


Figura A.2: Estructura de subdirectorios del directorio: */scripts*.

Apéndice B

Instalación y Versiones

B.1. Versiones de las herramientas utilizadas

En esta sección del apéndice se recogen las versiones de las herramientas utilizadas para llevar a cabo el desarrollo del código, así como las librerías utilizadas (ver Tabla B.1), de tal forma que los scripts proporcionados puedan ser ejecutados correctamente sobre estas. Cabe destacar toda la implementación de código ha sido desarrollada en el lenguaje de programación Python en su versión 3.7, por lo que su funcionamiento en versiones anteriores, como la 2.X o en las versiones inferiores de la rama 3.X no está garantizada.

Herramienta/Librería	**Versión Utilizada**
Anaconda	3
Dash	1.12.0
Git	2.25.1
Jupyter Notebook	6.0.3
Matplotlib	3.1.3
Numpy	1.18.1
Pandas	1.0.3
Python	3.7.0
Scikit-learn	0.22.1
Seaborn	0.10.0

Tabla B.1: Versiones de las herramientas y librerías utilizadas.

B.2. Instalación de la herramienta

A continuación se presentan las instrucciones para llevar a cabo la instalación tanto de la herramienta de visualización como la ejecución de los scripts adjuntos en la entrega de este proyecto.

1) **Instalación de Python:** Para llevar a cabo la instalación de Python, hay que descargar e instalar el ejecutable correspondiente de la siguiente página: https://www.python.org/downloads/ y seguir las indicaciones proporcionadas en el instalador (durante el proceso

de instalación es muy recomendable marcar la opción de **"Añadir Python a la variable PATH"** y la opción **"Instalar pip"**).

2) **Instalación de las librerías necesarias:** Una vez descargada e instalada la distribución de Python, es el momento de instalar las librerías necesarias para ejecutar los scripts y el *dashboard* correctamente. Para ello, se utilizará el gestor de paquetes *pip* (el cual viene incluido en la instalación de Python) y el fichero ***requirements.txt*** incluido en la entrega del proyecto. Para llevar a cabo esta instalación hay que abrir una terminal en el mismo directorio donde se encuentre el fichero *requirements.txt* y ejecutar el siguiente comando:

```
pip install -r ./requirements.txt
```

Una vez completado, se habrán instalado todas las librerías necesarias en sus versiones correspondientes para poder ejecutar de forma correcta tanto los scripts del proyecto como la herramienta de visualización.

3) **Ejecución de los scripts:** Para ejecutar un script, hay que acceder desde la terminal al directorio donde se encuentre alojado el script a probar y ejecutar el siguiente comando:

```
python <nombre_del_script.py>
```

4) **Ejecución de la herramienta:** Para ejecutar la herramienta, hay que acceder al directorio del proyecto /**app** y ejecutar el siguiente comando:

```
python app.py
```

Una vez hecho esto deberíamos ver algo similar a la Figura B.1.

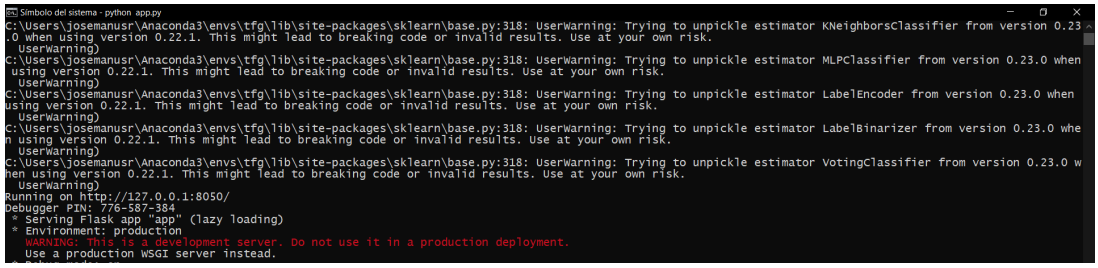


Figura B.1: Salida tras la ejecución del *dashboard*.

Finalmente, para visualizar e interactuar con el *dashboard* hay que abrir el navegador y acceder a la siguiente dirección: http://localhost:8050/.

Apéndice C

Manual de Usuario

En esta sección se presenta un manual de usuario en el que se detallan las funcionalidades de la herramienta de visualización e interacción desarrollada para que cualquier usuario tenga conocimiento sobre cómo hacer uso de la misma. El acceso a la misma se realiza mediante un navegador web, ya que esta se despliega de manera local en el ordenador. Cabe destacar que debido a que el objetivo principal consiste en presentar de forma visual tanto los resultados obtenidos como la interacción con el modelo de aprendizaje automático desarrollado, no se ha implementado ningún mecanismo de autenticación a la hora de hacer uso de la misma. La funcionalidad del *dashboard* se encuentra distribuida en tres pestañas, de tal forma que en cada una de ellas se realice una funcionalidad concreta. Estas pestañas son:

1) **Datos:** Es la pestaña principal, en ella se puede ver una tabla con la información de los vuelos registrados, así como una visualización de la distribución de pistas del Aeropuerto de Barajas y sus distintas configuraciones.

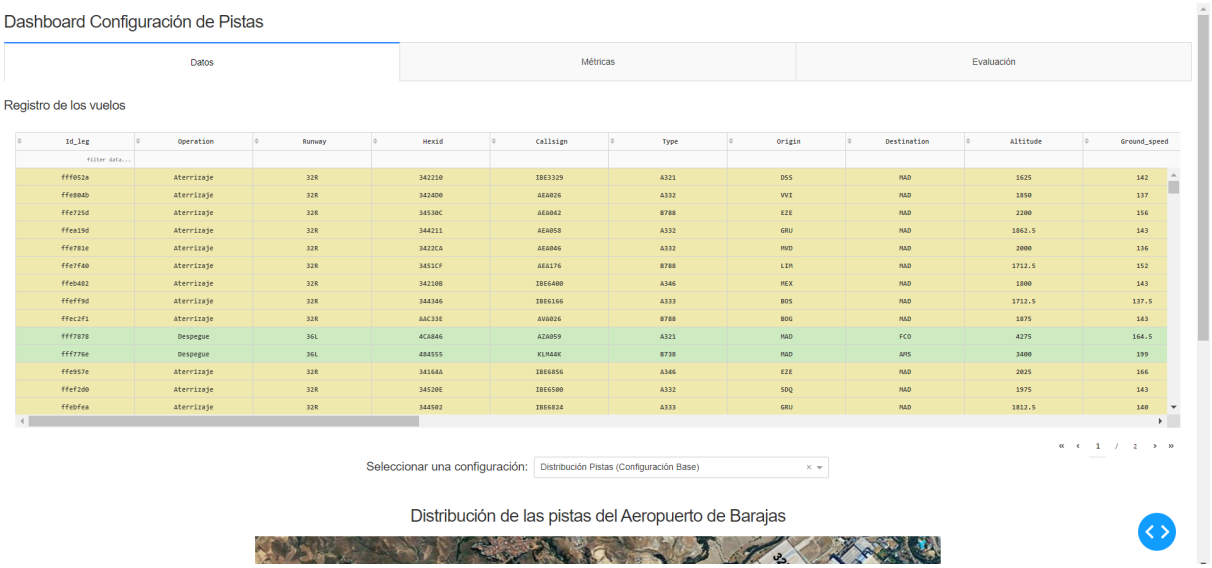


Figura C.1: *Dashboard*: Pestaña de datos.

2) **Métricas:** Esta pestaña permite visualizar las diferentes métricas que se han obtenido para los

modelos utilizados durante el desarrollo del proyecto. Para ello utiliza un gráfico interactivo en el que es posible ver con todo detalle cada uno de los valores obtenidos, así como la representación de los mismos en forma de tabla. Las métricas que se encuentran disponibles son: *Precision*, *Curva de Validación*, *Matriz de Confusión* y *Validación Cruzada*. Por otra parte, los modelos para los cuales se disponen estas métricas son: *Bosques Aleatorios*, *K-Vecinos Más Cercanos*, *Redes Neuronales* y *Clasificación por Votación*, cuya configuración de hiperparámetros es la utilizada una vez se completó el proceso de refinamiento de los modelos.

Figura C.2: *Dashboard*: Pestaña de métricas.

3) **Evaluación:** El objetivo de esta pestaña consiste en hacer de nexo entre el modelo final y el usuario. Para ello, consta de un pequeño formulario de entrada de datos en el que el usuario puede introducir manualmente las características de un determinado vuelo para, posteriormente, transferir esos datos al modelo y que este realice la predicción de la configuración de pistas a utilizar en función de los datos introducidos. Finalmente una vez el modelo determina la configuración en función de dichos datos, muestra por pantalla un gráfico interactivo con la distribución de las pistas para dicha configuración, pudiendo observar cuáles de ellas se utilizan y qué operación se realiza sobre estas.

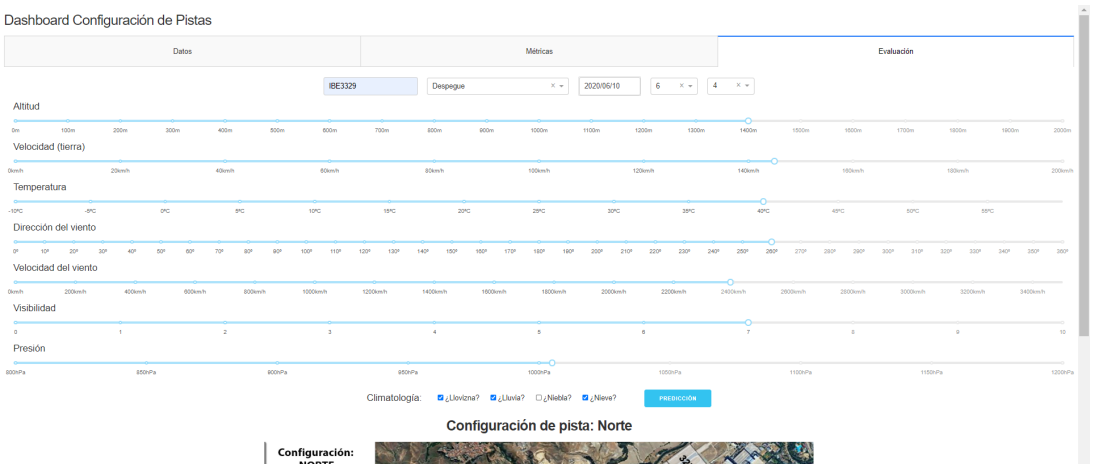


Figura C.3: *Dashboard*: Pestaña de evaluación.

Glosario

A

Aprendizaje Automático:

Proceso por el cual se le proporciona a un ordenador la capacidad de aprender continuamente sin necesidad de haber sido programados explícitamente para ello.

Autoridades Conjuntas de Aviación:

Máximo organismo de regulación de la aviación civil dentro del espacio aéreo europeo.

C

Caja Blanca:

Conjunto de pruebas realizadas a una herramienta software para verificar el correcto funcionamiento de este a nivel interno. Estas pruebas reciben este nombre porque en todo momento se conoce el funcionamiento interno de la funcionalidad que se está probando.

Caja Negra:

Conjunto de pruebas realizadas a una herramienta software para verificar el correcto funcionamiento de este a nivel externo. Para este tipo de pruebas únicamente se analiza la salida obtenida a partir de una determinada entrada y se comprueba sí es o no es la esperada, independientemente de su funcionamiento interno.

Capacidad Operativa:

Número de operaciones de aterrizaje y de despegue por hora que es capaz de albergar un aeropuerto. Este valor varía en función del número de pistas que disponga el aeropuerto, su orientación, su estado, etc.

Cielo Único Europeo:

Iniciativa llevada a cabo por la Unión Europea en 2004 con el objetivo de unificar el espacio aéreo europeo, mejorando la infraestructura de los aeropuertos y aeronaves, haciendo posible la gestión del creciente número de operaciones aéreas de los últimos años, mejorar tanto la seguridad y disminuir el impacto medioambiental.

Codifiación:

Asignación de valores numéricos a los distintos datos de una característica, de tal forma que se genere una nueva representación de esta característica con distintos valores pero con el mismo significado.

Codificación en caliente:

Técnica de normalización de datos por la cual se generan tantas nuevas características como valores diferentes exista en la característica a codificar.

Control del Tráfico Aéreo:

Servicio proporcionado por los controladores aéreos en la terminal, cuya función consiste en dirigir, controlar y asesorar las operaciones llevadas a cabo por las aeronaves a través del espacio aéreo.

D

Dashboard:

Herramienta software de gestión de la información que permite la monitorización, análisis e interacción del usuario con los datos que se representan en él.

Dataset:

Colección de datos generalmente tabulada. En su versión más simple es el equivalente a una tabla en una base de datos o una única matriz de datos estadística.

E

Escalabilidad:

Capacidad que tiene un sistema informático para mantener su rendimiento a medida que aumenta el número de usuarios y transacciónes que se realizan en él. Existen dos tipos de escalabilidad: escalabilidad horizontal (aumentar el número de hardware) y escalabilidad vertical (mejorar las especificaciones del hardware).

G

Generalización:

Capacidad que tiene un modelo de aprendizaje automático para clasificar o predecir resultados a partir de datos con los que no ha sido entrenado, es decir, datos totalmente nuevos y que nunca "ha visto".

Gestión del Tráfico Aéreo:

La gestión del tráfico aéreo comprende todas aquellas acciones e interacciones que se llevan a cabo para ayudar a una aeronave a trasladarse de un aeródromo de origen a otro de destino de una forma segura.

H

Hiperparámetro:

Parámetros de configuración de los modelos de aprendizaje automático que permiten regular el proceso de aprendizaje.

M

METAR:

Estándar internacional en el campo de la aviación para enviar periódicamente la información meteorológica en los aeropuertos y aeródromos.

Modelo de Datos:

Representación de una estructura abstracta que determina la estructuración y organización de un conjunto de datos, facilitando el entendimiento y la comprensión de los datos que la conforman.

Modelo:

Representación matemática de un sistema o de una realidad compleja.

N

NextGen:

Proyecto liderado por la Administración Federal de la Aviación, cuyo objetivo consiste en llevar a cabo una modernización en las tecnologías del espacio aéreo estadounidense, con la finalidad de mejorar la seguridad y la eficiencia en los vuelos, así como reducir sus costes.

Normalización:

Conjunto de operaciones de ajuste realizadas sobre los datos que permite su representación con respecto a una escala común.

O

Outlier:

Observaciones que se encuentran muy alejadas del conjunto de datos principal. Este tipo de observaciones pueden deberse a una medición errónea (si ha sido obtenido directamente) o a un error de ejecución (si ese valor es derivado de otro).

P

Proveedor de Servicios de Navegación Aérea:

Organización o empresa encargada de proporcionar servicio y administración (tanto en tierra como en aire) a las aeronaves. Algunos proveedores de Servicios de Navegación Aérea son: ENAIRE (España) o la Administración Federal de la Aviación (Estados Unidos).

R

Raw Data:

Datos recopilados en su forma original y sin haber aplicado ningún tipo de operación sobre ellos.

Reglas de Vuelo Instrumental:

Conjunto de normas en las que se regula el vuelo de las aeronaves utilizando aparatos y sistemas de visión asistida.

Reglas de Vuelo Visual:

Conjunto de normas y condiciones por las cuales el piloto se encuentra autorizado a utilizar la aeronave, sin la necesidad utilizar aparatos y sistemas de visión asistida.

Requisito de Usuario:

Representación de una funcionalidad requerida por el usuario que ha de satisfacer el sistema desarrollado.

Requisito Funcional:

Representación de una funcionalidad de un sistema software o de alguna de sus componentes.

S

SESAR:

Proyecto llevado a cabo por la Unión Europea en 2007, con el objetivo de construir y poner en práctica una política común para la gestión del tráfico aéreo en todo el espacio aéreo europeo, mejorando el rendimiento y la eficiencia de la acutal red de tráfico aéreo.

V

Variable Cualitativa:

Tipo de variable estadística que expresa cualidades o características de un objeto o persona. Este tipo de variables se caracterizan por expresar una cualidad o característica que no puede medirse numéricamente, por lo que no otorga datos específicos ni un orden.

Variable Cuantitativa:

Tipo de variable estadística que otorgan como resultado un valor numérico, haciendo posible su ordenación, representación y comparación. Un ejemplo de este tipo de variables es el peso o la altura.

Vigilancia Dependiente Automática-Contrato:

Tecnología utilizada para transmitir de forma abierta, autónoma, dependiente y periódica información de posicionamiento entre una aeronave y un Proveedor de Servicios de Navegación Aérea, en unas determinadas condiciones (contrato).

Vigilancia Dependiente Automática-Difusión:

Tecnología utilizada para transmitir de forma abierta, autónoma, dependiente y periódica información de posicionamiento de la aeronave a todos los receptores que se localicen dentro del radio de alcance de esos mensajes.

Vigilancia Dependiente Automática:

Tecnología de control del tráfico aéreo que permite a las aeronaves enviar de forma automática, la información de los sistemas de navegación que dispone a bordo.

Acrónimos

Acrónimo	Español	Inglés
ADS	Vigilancia Dependiente Automática.	Automatic Dependent Surveillance.
ADS-B	Vigilancia Dependiente Automática-Difusión.	Automatic Dependent Surveillance-Broadcast.
ADS-C	Vigilancia Dependiente Automática-Contrato.	Automatic Dependent Surveillance-Contract.
ANSP	Proveedor de Servicios de Navegación Aérea.	Air Navigation Service Provider.
ATC	Control del Tráfico Aéreo.	Air Traffic Control.
ATM	Gestión del Tráfico Aéreo.	Air Traffic Management.
CU	Caso de Uso.	Use Case.
IFR	Reglas de Vuelo Instrumental.	Instrument Fight Rules.
JAA	Autoridades Conjuntas de Aviación.	Joint Aviation Authorities.
KNN	K-Vecinos Más Cercanos.	K-Nearest-Neighbor.
METAR	Informe Meteorológico de Aeródromo.	Meteorological Aerodrome Report.
NextGen	Sistema de Transporte Aéreo de Próxima Generación.	Next Generation Air Transportation System.
RF	Requisito Funcional.	Functional Requirement.
RU	Requisito de Usuario.	User Requirement.
SES	Cielo Único Europeo.	Single European Sky.
SESAR	Cielo Único Europeo Investigación de la Gestión del Tráfico Aéreo.	Single European Sky ATM Research.
VFR	Reglas de Vuelo Visual.	Visual Flight Rules.