



**Universidad
Europea**

UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

GRADO EN INGENIERÍA MATEMÁTICA APLICADA AL

ANÁLISIS DE DATOS

TRABAJO FIN DE GRADO

ADOPTION MOVEMENT

LUCÍA NÚÑEZ ALONSO

CURSO 2022-2023

TÍTULO: ADOPTION MOVEMENT

AUTOR: LUCÍA NÚÑEZ ALONSO

TITULACIÓN: GRADO EN INGENIERÍA MATEMÁTICA APLICADA AL ANÁLISIS DE DATOS

DIRECTOR DEL PROYECTO: ANA DEL VALLE CORRALES PAREDES

FECHA: JUNIO de 2023

RESUMEN

En la actualidad, existen diversas aplicaciones y páginas web cuya finalidad principal se basa en fomentar las adopciones de animales centralizando la información de estos en una sola plataforma o bien agilizando telemáticamente los procesos de adopción.

Adoption Movement se centra en contribuir a la adopción efectiva de animales, pero poniendo el foco del problema en la saturación de las instalaciones de los centros de protección, es decir, además de pretender con la solución propuesta que los animales en adopción encuentren su hogar, se considera la importancia de destacar que no en todos los lugares se adoptan los mismos perfiles, y por eso se debe valorar el centro al que trasladar o ingresar un animal observando diferentes aspectos, no solamente la cercanía de las instalaciones.

Para la solución al problema, se ha desarrollado una aplicación móvil con la finalidad de visualizar las funcionalidades de la misma, basadas en diferentes modelos de inteligencia artificial y complejas técnicas para la obtención de datos, entre las que están recomendar animales compatibles con adoptantes en base a un test realizado por los últimos, y centralizar la información de centros de protección animal españoles para que esta pueda ser perfectamente accesible por adoptantes o miembros de otros centros.

Palabras clave: saturación, protectora, animal, adopción.

ABSTRACT

Currently, there are several applications and web pages whose main purpose is to promote the adoption of animals by centralizing their information on a single platform, or by telematically streamlining the adoption processes.

Adoption Movement, focuses on contributing to the successful adoption of animals, but putting the focus of the problem on the saturation of the facilities of the protection centers, that is, in addition to intending with the proposed solution that the animals for adoption find their home, it is considered the importance of highlighting that not in all places the same profiles are adopted, and that is why the center to which to move.

For the solution to the problem, a mobile application has been developed in order to visualize its functionalities, based on different artificial intelligence models and complex techniques for obtaining data, which also contains other additional functionalities such as recommending animals that are compatible with adopters, this being based on a test carried out by the latter, and centralizing the information of Spanish animal protection centers so that it can be perfectly accessible by adopters or members of other centers.

Key words: saturation, protection, animal, adoption.

AGRADECIMIENTOS

A mi tutora Ana del Valle, en primer lugar, por haberme aceptado como tutelada a pesar de no pertenecer al grado de Ingeniería Informática, por acoger el tema propuesto y por ayudarme a desarrollarlo aportándome su constante apoyo y enseñanza.

A mi familia y en especial a mi madre, quien hizo todo lo posible para que pudiese superar exitosamente todos y cada uno de mis logros, por ser mi pilar fundamental en la vida académica y por haberme enseñado las claves de la organización, disciplina, esfuerzo y constancia.

A mi amiga Andrea Gil, filóloga inglesa, por haberme ayudado con cualquier duda relacionada con la lengua inglesa, por haberme ayudado a entender y aprender conceptos necesarios en este proyecto y durante los años de universidad, y por haberme dado la idea del nombre “Adoption Movement”.

A mi amigo y compañero Andrés Fernández, por haberse prestado a cualquier tipo de ayuda en todo momento, por haberme orientado a dar los primeros pasos en el funcionamiento general del Backend, y por haber colaborado conmigo en todos los proyectos y trabajos de la universidad del primer al último día.

A mi pareja Pablo Taboada en lo personal, por haber fomentado la confianza sobre mí misma, por la paciencia y la calma que me ha aportado durante la realización del proyecto, y por animarme a seguir adelante a pesar de los inconvenientes.

A todos aquellos que me han acompañado a lo largo de estos años, tanto en la vida personal como en la académica.

Gracias.

TABLA RESUMEN

Nombre y apellidos	Lucía Núñez Alonso
Título del proyecto	Adoption Movement
Directores del proyecto	Ana del Valle Corrales Paredes
¿El proyecto ha implementado un producto?	SÍ
¿El proyecto ha consistido en el desarrollo de una investigación o innovación?	SÍ
Objetivo general del proyecto	Aplicación móvil que relaciona animales con centros donde según sus características serán adoptados con mayor probabilidad.

Índice

RESUMEN	4
ABSTRACT	4
TABLA RESUMEN	6
Capítulo 1. INTRODUCCIÓN	11
1.1 Contexto y justificación	11
1.2 Planteamiento del problema	11
1.3 Propuesta de proyecto	12
1.4 Objetivo general	12
1.5 Objetivos específicos	12
1.6 Beneficios del proyecto	13
1.7 Estructura de la memoria	13
Capítulo 2. ESTADO DEL ARTE	15
2.1 Aplicaciones relacionadas	15
2.1.1 Guau! Qué perros	15
2.1.2 AdoptaMe	15
2.1.3 Miwuki	16
2.1.4 GetPet	17
2.2 Conocimiento adquirido en el área de desarrollo	17
2.2.1 Información para el recomendador de centros por animal:	17
2.2.2 Información para la relación adoptantes-animales:	18
2.2.3 Información para centralizar los datos de las diferentes asociaciones:	19
2.3 Datos utilizados	19
2.3.2 Datos de centros	19
2.3.3 Datos de animales	21
2.4 Tecnologías utilizadas	22
2.4.1 Obtención, limpieza y tratamiento de datos	22
2.4.2 Generación de datos sintéticos	23
2.4.3 Algoritmos de Inteligencia Artificial	23

2.4.4	Desarrollo del Backend.....	23
2.4.5	Desarrollo del Frontend	23
2.4.6	Otras.....	24
Capítulo 3.	DESARROLLO DEL PROYECTO	25
3.1	Planificación del proyecto	25
3.1.1	Formación en el ámbito en el que se desarrolla el proyecto – FASE 1	25
3.1.2	Obtención, limpieza y tratamiento de datos – FASE 2.....	25
3.1.3	Algoritmos de Inteligencia Artificial – FASE 3	26
3.1.4	Desarrollo del Backend – FASE 4.....	26
3.1.5	Desarrollo del Frontend – FASE 5.....	26
3.1.6	Documentación y presentación – FASE 6.....	26
3.2	Presupuesto	27
3.3	Desarrollo del proyecto.....	28
3.3.1	Formación en el ámbito en el que se desarrolla el proyecto – FASE 1	28
3.3.2	Obtención, limpieza y tratamiento de datos – FASE 2.....	29
3.3.3	Algoritmos de Inteligencia Artificial – FASE 3	49
3.3.4	Desarrollo del Backend – FASE 4.....	56
3.3.5	Desarrollo del Frontend – FASE 5.....	58
Capítulo 4.	DISCUSIÓN.....	61
Capítulo 5.	CONCLUSIONES	62
5.1	Conclusiones del trabajo	62
5.2	Conclusiones personales	62
Capítulo 6.	FUTURAS LÍNEAS DE TRABAJO	64
Capítulo 7.	GUÍA DE LA APLICACIÓN	66
	BIBLIOGRAFÍA.....	72
	ANEXOS	78

Índice de Figuras

Figura 1: *Diagrama de Gantt sobre la planificación del proyecto*

Figura 2: *Comparación de frecuencias de muestra y población sintética*

Figura 3: *Superposición de frecuencias de muestra y población sintética*

Figura 4: *Porcentajes iniciales de la distribución de una muestra*

Figura 5: *Porcentajes finales de la distribución de una población sintética*

Figura 6: *Diagrama ER de los conjuntos de datos*

Figura 7: *Diagrama de arquitectura del proyecto*

Figura 8: *Diagrama de casos de uso*

Índice de Tablas

Tabla 1: *Librerías utilizadas para el proceso de Web Scraping*

Tabla 2: *Librerías utilizadas para la generación de datos ficticios*

Tabla 3: *Datos de centros de protección animal*

Tabla 4: *Datos de animales en estado de adopción*

Tabla 5: *Datos de animales adoptados*

Tabla 6: *Tipos de inicialización en clustering*

Tabla 7: *Tipos de métricas en filtrado colaborativo*

Tabla 8: *Presupuesto del proyecto*

Capítulo 1. INTRODUCCIÓN

1.1 Contexto y justificación

“Nuestro país se sitúa desde hace años a la cabeza de Europa en cifras de abandono animal. La falta de concienciación y la dejadez política ponen en jaque un sistema de protectoras completamente saturado.” [1]

(Crespo Garay, 2021)

Las cifras españolas de abandono animal están aumentando año tras año, lo cual se traduce directamente a un problema para los centros de protección animal y otras entidades encargadas de dar cobijo a las diferentes especies sin hogar.

Debido a la escasez de recursos con las que cuentan estas organizaciones sin ánimo de lucro, la saturación de las instalaciones con las que cuentan para desempeñar sus labores es cada vez más preocupante, implicando directamente un riesgo para la salud y el bienestar de los animales y de las personas a cargo de ellos

1.2 Planteamiento del problema

El proyecto propuesto, cubre diferentes necesidades que en conjunto facilitan el funcionamiento de los centros de protección animal de España.

En primer lugar, la aplicación desarrollada se enfoca en evitar la saturación de las instalaciones de cada una de las asociaciones tratando de disminuir la ocupación de las mismas proponiendo para cada animal centros donde este mismo en base a sus características pueda ser más probablemente adoptado.

En segundo lugar, Adoption Movement proporciona una herramienta que relaciona adoptantes con sus animales compatibles mediante un cuestionario con la finalidad principal de crear vínculos efectivos disminuyendo las posibilidades de que los animales puedan ser devueltos o de que la relación humano-mascota no sea tan agradable.

En la actualidad, son los recursos humanos de las protectoras quienes entrevistan a cada candidato a adoptante para analizar su compatibilidad con el animal de interés, la solución propuesta automatizaría completamente este paso, pudiendo permitir una gran optimización de tiempo de los voluntarios de los centros.

Por último, la aplicación permite la accesibilidad a información de todas las asociaciones de protección animal españolas de manera centralizada y organizada. Estos datos no constaban de este modo en ninguna aplicación ni plataforma, por ello ha sido uno de los objetivos principales del proyecto.

1.3 Propuesta de proyecto

La aplicación Adoption Movement (“Movimiento de adopción”), recibe este nombre en base a su objetivo principal: colaborar en conjunto con los centros de protección animal para la adopción efectiva de los animales presentes en sus instalaciones evitando llegar al punto de la saturación de las mismas, y en caso de llegar a tal punto, proponer otros centros donde cada animal específico tendrá una mayor probabilidad de ser adoptado.

Para alcanzar el objetivo principal, se ha implementado una aplicación móvil de manejo lo más sencillo e intuitivo posible para el usuario, orientada a miembros de centros de protección animal fundamentalmente, o a particulares que deban deshacerse de sus animales de compañía, o por lo contrario busquen una nueva mascota.

Dos de los puntos más importantes del proyecto, además de la aplicación móvil desarrollada para visualizar los resultados del mismo, han sido la consecución de los datos utilizando diferentes técnicas como *Web Scrapping* o *Synthetic Data*, y por otro lado la programación de los algoritmos de Inteligencia Artificial para cada una de las funcionalidades, ambos puntos constan explicados en profundidad en apartados posteriores de la memoria.

1.4 Objetivo general

La creación y desarrollo de una aplicación móvil con el mismo nombre que el proyecto, que reúna varias funcionalidades relacionadas con la finalidad de contribuir a solucionar el problema de la saturación de los centros de protección animal españoles, principalmente sugiriendo para cada ejemplar en concreto, centros en los que este podría tener una mayor posibilidad de ser adoptado y secundariamente, relacionando usuarios de la plataforma con animales en adopción en base a las características de ambos.

1.5 Objetivos específicos

- a. Obtención de los datos mediante técnicas de *web scraping* o *synthetic data*, siguiendo el proceso ETL (Extract, Transform, Load).
- b. Obtención de características mediante técnicas de procesamiento de texto, de lenguaje natural y análisis de sentimiento.
- c. Programación de un cluster de tipo kmodes con la finalidad de recomendar centros para cada animal en concreto tras aplicar funciones a la salida del algoritmo.
- d. Programación de un algoritmo de filtrado colaborativo basado en ítems con la finalidad de relacionar respuestas de un cuestionario realizado por un usuario adoptante, y animales con los que estas sean compatibles.
- e. Implementación de un servidor utilizando *FastApi* en *Python*.
- f. Programación de funciones de *backend* para adaptar los datos de salida de los algoritmos a los deseados.
- g. Programación y diseño de *frontend* utilizando *Android Studio*, su framework Flutter y su lenguaje de programación propio *Dart*.
- h. Definición de funciones en *Dart* para interaccionar con el servidor.

- i. Puesta en práctica de los puntos más relevantes aprendidos durante el grado universitario Ingeniería Matemática Aplicada al Análisis de Datos.
- j. Aprendizaje autodidacta de nuevas herramientas interesantes para el proyecto.
- k. Control de versiones del proyecto

1.6 Beneficios del proyecto

La Inteligencia Artificial y sus correspondientes aplicaciones, cada vez están más presentes en el día a día aportando sus beneficios de manera muy efectiva. Esta aplicación transporta estos beneficios a un ámbito con recursos limitados como es el de la protección animal en España.

Por un lado, optimiza el tiempo de los usuarios y también de los miembros de protección animal, agiliza la búsqueda de centros óptimos para animales concretos sin necesidad de invertir horas de en ello, relaciona animales compatibles con adoptantes sin tener que comprometer en esta tarea a alguna persona física miembro voluntario de la asociación, y por último, centraliza la información de los centros españoles tales como protectoras o refugios de animales en una sola página, facilitando el contacto de unos con los otros, y la accesibilidad al usuario a datos de interés.

Diferentes técnicas complejas de obtención de datos han hecho posible el desarrollo del prototipo de la aplicación a pesar de las dificultades encontradas para la utilización de datos reales para el proyecto.

Además, la interfaz gráfica de la aplicación se caracteriza por ser sencilla e intuitiva para que pueda ser utilizada sin complicación por el mayor número de público posible.

1.7 Estructura de la memoria

El contenido de la siguiente memoria se divide en los siguientes 8 capítulos:

Capítulo 1. INTRODUCCIÓN: Descripción del objetivo, el contexto y la problemática que motivaron la realización del proyecto, con lo que se pretende proporcionar una visión general del proyecto, su propuesta, objetivos y beneficios.

Capítulo 2. ESTADO DEL ARTE: Explicación detallada de los datos y tecnologías empleadas, de los conocimientos adquiridos en el área de desarrollo previa elaboración del proyecto y sus correspondientes fuentes de información; y análisis de las diferentes aplicaciones relacionadas.

Capítulo 3. DESARROLLO DEL PROYECTO: Descripción de cada una de las fases superadas durante la realización del proyecto en base a la previa planificación de este.

Capítulo 4. DISCUSIÓN: Reflexión de los resultados obtenidos y del desarrollo del proyecto.

Capítulo 5. CONCLUSIONES: Planteamiento de conclusiones obtenidas tras la finalización del proyecto a nivel técnico y a nivel personal.

Capítulo 6. FUTURAS LÍNEAS DE TRABAJO: Descripción de posibles tareas futuras a realizar dentro del mismo proyecto, así como mejoras de este.

Capítulo 7. GUÍA DE LA APLICACIÓN: Manual de utilización de la aplicación dedicado al usuario.

Capítulo 2. ESTADO DEL ARTE

En este capítulo, se analizarán los avances más recientes y relevantes relacionados con la aplicación a desarrollar con la finalidad de dar a conocer la situación que tendría el proyecto en el contexto de la actualidad. Además, se hace referencia detallada a la información adquirida en el área de estudio previa realización del proyecto describiendo también su procedencia, así como la de los datos utilizados para el desarrollo de la aplicación. Por último, en este apartado de la memoria se enumeran las tecnologías utilizadas durante el proyecto y la finalidad del uso de cada una de ellas.

2.1 Aplicaciones relacionadas

Adoption Movement, nace como se ha mencionado anteriormente, a partir de una o varias necesidades no cubiertas en el ámbito de la protección y el bienestar animal. Esto implica que no existe ni siquiera fuera de España, ninguna aplicación web ni móvil que realice las mismas funcionalidades y ni siquiera algunas similares a las que ofrece este proyecto, lo cual es gran parte de su complejidad.

A pesar de que el área de la protección animal sin ánimo de lucro se caracteriza por la escasez de recursos que limitan el presupuesto de las soluciones a la voluntad de los desarrolladores, existen aplicaciones relacionadas sobre todo con la visibilidad de los animales en adopción de los centros españoles. A continuación, se muestran algunos casos:

2.1.1 Guau! Qué perros

“App que podrán usar protectoras y usuarios. Las protectoras podrán poner fotos de sus perritos y clasificarlos en tres categorías: perdidos, abandonados o encontrados. Luego los usuarios podrán hacer buscando y filtrando el tipo de perro, el color, la ciudad, la zona, etc.” [34]

(Editorial, 2022)

Esta aplicación, busca una solución a las adopciones efectivas de animales poniendo el foco en la difusión de cada caso. Del mismo modo que Adoption Movement, está adaptada para que pueda ser utilizada por usuarios particulares o bien miembros de protección animal, pero el objetivo del proyecto no se centra tanto en la disminución de la ocupación de las instalaciones de los centros si no en la búsqueda de hogar de animales sin ellos sea cual sea su situación (adopción, pérdida etc.). Permite aplicar filtros a los resultados en base a diferentes características tanto de ubicación como correspondientes al físico de los animales. [35]

2.1.2 AdoptaMe

“Está hecha con la idea de generar un punto único de adopciones en la sociedad (como un Trivago para adopción animal), así será mucho más fácil dar visibilidad y encontrar un nuevo hogar para muchos animales” [36]

Aplicación móvil para Android, cuyo objetivo es centralizar la información de los animales en adopción de toda España, y que del mismo modo que Adoption Movement, ha partido de un proyecto para un Trabajo de Fin de grado, pero del grado de Ingeniería Informática.

La aplicación es completamente gratuita y sin ánimo de lucro, lo cual reafirma la escasez de fuentes de ingresos posibles para este tipo de aplicaciones benéficas.

AdoptaMe, permite por otra parte, la elaboración de un perfil de usuario bastante completo para que los centros de protección animal puedan tener una primera imagen de los posibles adoptantes, y también ponerse en contacto con ellos. [37]

Por otro lado, la utilización de la aplicación está destinada a usuarios de protectoras y a usuarios adoptantes, diferenciando cada perfil mediante el registro y el inicio de sesión a la misma.

2.1.3 Miwuki

Programa de gestión de centros de protección animal gratuito, que además aporta la funcionalidad de facilitar el contacto entre los mismos y usuarios interesados en la adopción o acogida de los animales.

Fue creado por un Ingeniero Informático y un Veterinario, y es una de las soluciones digitales más completas en el ámbito de desarrollo del actual proyecto. Dentro de la funcionalidad de gestión de centros, permite que los miembros puedan tener un perfil donde figuren los datos de acogidas, adopciones, vacunaciones y desparasitaciones de los animales entre otros; dentro de la otra funcionalidad, vincular animales con adoptantes, permite realizar un test de manera similar a la que lo hace Adoption Movement, para relacionar las preferencias y hábitos de los usuarios con animales que se adapten a las mismas. [38]

El formulario mencionado [39], desarrollado por el programa Miwuki con el apoyo de la Cátedra Fundación Affinity de la UAB, consta de once preguntas relacionadas con los gustos sobre animales y el estilo de vida de los adoptantes, y la principal diferencia entre esta funcionalidad y la propuesta por Adoption Movement, será la presentación de los resultados del test. Mientras que Miwuki muestra solamente los animales que coinciden con los criterios exactos proporcionados (el resultado puede ser nulo), Adoption Movement proporcionará un listado más amplio y por tanto con mayor margen en todos los aspectos, que atribuya al usuario la capacidad de decidir posteriormente. En el test de Adoption Movement, no será tan tenido en cuenta el aspecto del animal ya que esto se considera secundario, los adoptantes podrán seleccionar el animal que más les atraiga una vez que se muestren los resultados, los cuales se podrán afinar posteriormente en base a atributos relacionados con el animal y su localización geográfica. Por otra parte, se aportará información de cada uno de los resultados en relación con el estilo de vida que requiere el animal o bien al que está acostumbrado.

2.1.4 GetPet

La única de las aplicaciones descritas que no es de origen español. Se trata de una solución basada en el funcionamiento de la famosa aplicación de citas *Tinder* [40], donde cada animal tiene su perfil creado con fotografías y una descripción, y el usuario adoptante puede deslizar a la derecha si este ha llamado su atención, o bien a la izquierda para continuar buscando entre el resto de los animales.

La aplicación fue inventada en Lituania, y solamente permite facilitar adopciones caninas, el resto de las especies no se encuentran dentro del alcance de la solución por el momento. [41]

Como conclusión a las aplicaciones relacionadas con Adoption Movement, se obtiene la recalcada necesidad de crear una solución enfocada en la solución a la saturación de las instalaciones. Por ello una de las funcionalidades del proyecto actual, permite recomendar centros en base a su registro histórico de adopciones para cada animal introducido, planteando la opción de poder trasladar el animal a dicho lugar y hacer que su adopción se cumpla con mayor probabilidad.

2.2 Conocimiento adquirido en el área de desarrollo

Partiendo de la familiarización con el mundo del abandono animal, los centros de protección, las adopciones y acogidas y todo lo relacionado con ello, se ha contactado con la colaboración de varios miembros de diferentes centros de protección animal españoles para aclarar aspectos relacionados con situaciones y procedimientos reales que permitieran ajustar a estos las funcionalidades de la aplicación en la mayor medida posible.

Conviene aclarar para cada diferente funcionalidad principal de la aplicación, la información recopilada para cada caso:

2.2.1 Información para el recomendador de centros por animal:

Las leyes españolas que soportan temas relacionados con el abandono animal o la protección de los animales de compañía, varían en función de la Comunidad Autónoma en cuestión, pero en términos generales el actual informe se refiere a centros de protección animal (o “protectoras”), como organizaciones sin ánimo de lucro que en ningún caso sacrifican animales por temas de saturación de las instalaciones, y a centros Municipales de Acogida (o “perreras”), como empresas privadas contratadas por ayuntamientos para asegurar que en cada municipio se cumplan las leyes del bienestar animal correspondientes.

Mientras que un animal puede permanecer en las instalaciones de una protectora desde el día en el que ingresa hasta el día en el que fallece de manera natural si nadie lo ha adoptado antes, en el caso de las perreras una vez que ingresa dicho animal, se espera que este sea reclamado por su dueño en un plazo aproximado de 10 días, de no ser así el animal pasa a figurar en estado de adopción durante cierto plazo establecido por cada municipio (generalmente 21 días), y si

tras el vencimiento de dicho plazo el animal en cuestión no ha sido adoptado, este será sacrificado como remedio rápido y eficaz para combatir la saturación de las instalaciones.

Adoption Movement, se centra en colaborar con centros de protección animal de manera general, quienes mayoritariamente sufren el problema de la saturación de sus instalaciones. Además, esta herramienta podrá ser utilizada por particulares que necesiten por cualquier motivo, dar en adopción a sus animales de compañía.

La utilidad de la aplicación de recomendar centros funciona de manera que las características de un animal en concreto introducidas por el usuario de la aplicación sean la entrada de datos para un modelo de Inteligencia Artificial que en base a los datos introducidos, y a otros datos de animales adoptados en diferentes centros de protección animal españoles, genere un listado de centros recomendados para el animal en concreto, donde este tendrá una mayor probabilidad de ser adoptado.

Un miembro de la *Sociedad Protectora de Animales y Plantas de Lugo* [2], en una de las entrevistas realizadas, describió una situación vivida hace un par de años en la que Adoption Movement hubiera servido de gran ayuda. Siendo la ocupación máxima de las instalaciones de la asociación un número entre cien y doscientos, una finca en la provincia de la protectora fue descubierta con un centenar de perros abandonados en ella. Todos estos animales tuvieron que ser hospedados en las instalaciones de la Sociedad Protectora de Animales y Plantas de Lugo, estando estas diseñadas para alojar un solo animal por canil, pero el problema del gran número de animales abandonados habría desembocado en tener que repartir hasta tres perros por jaula, un riesgo para los animales, quienes tuvieron que sobrevivir en condiciones poco deseables hasta que los miembros de la protectora encontrasen otros centros a los que poder trasladar especies para poder controlar la ocupación de las instalaciones.

Por otro lado, durante la entrevista se preguntó el criterio a seguir en base el cual se eligen los animales que deben ser trasladados de centro en casos de urgencia como los descritos anteriormente; y es que no existe ningún criterio, los animales son elegidos “a dedo”, y la solución planteada fundamentaría cada posible traslado en base a las posibilidades que este podría tener de ser adoptado en el centro de destino.

2.2.2 Información para la relación adoptantes-animales:

Siguiendo con la información recopilada mediante entrevistas, se realizaron preguntas a miembros de diferentes centros de protección animal españoles en relación con el proceso a seguir previo a las adopciones o a los contratos de casa de acogida para poder concretar la funcionalidad de relacionar adoptantes con animales, basada en la elaboración de un cuestionario por parte del usuario, que devuelva un listado de animales con los que su estilo de vida sería compatible.

De los contactos mantenidos, se concluyó que el procedimiento a seguir era muy similar en la mayoría de los casos, y fue un miembro de la *Protectora Amics dels Animals dels Segrià* [3] quien detalló minuciosamente todos los pasos a realizar:

En primer lugar, el adoptante se interesaría por uno de los animales en concreto, y entonces los miembros del centro de protección animal encargados de contactar con personas externas, le harían una entrevista con la finalidad de concluir si este cumple o no con las necesidades rutinarias del animal. Las preguntas a realizar durante la entrevista responderían a aspectos

personales de la vida del interesado como por ejemplo número de residentes en su vivienda habitual, horas de trabajo diarias, nivel de actividad física, número de hijos etc. Si el resultado de la entrevista fuera apto y el interesado y el animal por tanto fueran compatibles, el siguiente paso sería tramitar la adopción o acogida, y de lo contrario, el interesado (si lo siguiera estando), volvería a escoger un animal con el que se volverían a contrastar los resultados de la entrevista, y así sucesivamente hasta terminar encontrando la compatibilidad. La solución aportada, además de automatizar este paso, genera para cada cuestionario un listado de animales entre los cuales el interesado puede escoger su nuevo animal de compañía; en este listado ya se incluyen únicamente los animales compatibles con el estilo de vida del usuario, lo cual supone un gran ahorro de tiempo partiendo de la base de que no existen comparaciones en vano entre animales e interesados no compatibles, además de no necesitar el tiempo de personal de la organización para realizar entrevistas ni comparaciones ya que las mismas son hechas por la aplicación de una manera inteligente.

2.2.3 Información para centralizar los datos de las diferentes asociaciones:

Un problema para resaltar en el mundo de los centros de protección animal es la falta de una base de datos completa que refleje la información de cada una de las asociaciones y sus correspondientes métodos de contacto de manera centralizada.

Adoption Movement, tendrá como otra de sus funcionalidades, un apartado en el que esta información estará organizada y será completamente accesible por el usuario.

2.3 Datos utilizados

Para el desarrollo de la aplicación, ha sido necesaria la obtención de datos relacionados con centros de protección animal, y de animales de dichos centros, a continuación, se explica el proceso de consecución de los datos, y la utilización de los mismos dentro del proyecto en cada caso:

2.3.2 Datos de centros

En primer lugar, se ha optado por tratar de encontrar alguna fuente de datos abiertos por comunidad autónoma o a nivel nacional que recogiese la información para poder reflejarla directamente en la aplicación en pasos posteriores, además se ha contactado por vía telefónica y o por correo electrónico con diferentes entidades y federaciones españolas encargadas del bienestar y los derechos animales.

El fracaso de la consecución de datos por las vías mencionadas anteriormente reafirmó la necesidad de incorporar la funcionalidad de centralizar los datos de los centros de protección animal en la aplicación.

Uno de los contactos telefónicos mencionados anteriormente, sugirió la idea de buscar los datos en el registro de asociaciones público de cada una de las Comunidades Autónomas españolas, solución que finalmente se llevó a cabo.

Cada Comunidad Autónoma tiene el deber de publicar un listado de asociaciones actualizado accesible para cualquier ciudadano que lo desee consultar, este listado en algunos casos fue exportado fácilmente a través de la descarga de un archivo Excel o CSV, pero en otros casos la extracción ha sido más compleja al tratarse de otros formatos como ODS o PDF, o directamente al no existir esta opción de exportación del listado.

Para pasos posteriores en el tratamiento y filtrado de los datos, el formato ideal hubiera sido cualquiera que pudiera ser importado sin modificaciones en un script de Python haciendo uso de la librería Pandas (por ejemplo, Excel o CSV), por ello en los casos en los que los datos se encontraron en diferentes formatos se han utilizado conversores para poder transformarlos.

En el caso de las Comunidades Autónomas que no facilitasen la opción de exportar los datos en el formato que fuera, estos han sido extraídos utilizando técnicas de *Web Scraping*, y de nuevo almacenándolos en estructuras de tipo DataFrame de la librería *Pandas* de Python.

Una vez conseguidos los datos de las diferentes asociaciones de cada una de las Comunidades Autónomas de España, estos han sido filtrados por medio de programas que analizasen la categoría a la que pertenecen en caso de figurar esta misma (salvando únicamente la información de categorías relacionadas con la protección animal), y en caso de no existir ninguna variable que determinase la categoría, el filtrado se ha realizado por medio de la elección de palabras claves que pudieran estar contenidas en los propios nombres de las asociaciones (protectora, animal, gato, perro...).

Posteriormente a la filtración de datos, se ha procedido al rellenado de valores faltantes considerados importantes para la aplicación (correo electrónico, número de teléfono, sitio web y página de Facebook de cada uno de los centros). Este rellenado se ha realizado haciendo uso de librerías de datos generados en Python como *Faker*, que se explicarán en profundidad en apartados siguientes de la memoria.

En otras circunstancias donde el tiempo no fuera la principal amenaza del proyecto, se valoraría el rellenado de estos datos a mano con la finalidad de que siendo reales cumpliesen el objetivo de centralizar información verídica para los usuarios de la aplicación. En ese caso, también se podría incorporar un logo por organización en la base de datos para conseguir un contenido más atractivo en la aplicación.

De cara a la posterior visualización de los datos en mapas, se planteó la necesidad de que cada uno de los registros correspondientes a cada protectora, contase con su latitud, y para ello se ha realizado un programa que con la ayuda de la librería *GeoPy* de Python, crease dos nuevas variables para cada registro correspondientes a la latitud y la longitud, a partir de la dirección (en cadena de caracteres) mediante geocodificadores.

De cara a la generación de datos sintéticos de animales adoptados y en adopción para cada uno de los centros (explicados en el siguiente punto), y para hacer más precisa la distribución de estos datos en los centros, se han generado códigos en base a los nombres de las asociaciones, de manera que si en este figuran palabras relacionadas con “felino” (gato, cat...) en pasos posteriores se pueda asociar a este centro una mayoría de animales de esta especie, y exactamente igual para “canino”. Todo lo demás sería interpretado como “mixto” ya que en el filtrado previo, se han descartado otro tipo de centros de protección animal tales como de animales exóticos, domésticos como equinos, cerdos, ovejas etc.

Importante aclarar que ha sido posible recopilar los datos de todas las Comunidades Autónomas exceptuando dos:

- **Galicia**, debido a que el listado de asociaciones no cuenta con opción de ser exportado, y además previa visualización de los datos en la web es necesario aceptar un CAPTCHA *Completely Automated Public Turing test to tell Computers and Humans Apart: test de Turing público y automático para distinguir a los ordenadores de los humanos* [4], lo cual dificulta en gran medida el programa de Web Scraping que hubiera sido la solución llevada a cabo también en otras Comunidades Autónomas.
- **Extremadura**, debido a que no cuenta con un listado público de su registro de asociaciones como el resto de las comunidades Autónomas, y, además, contactar con la Junta de Extremadura telefónicamente supone coste económico.

2.3.3 Datos de animales

Dada la limitación de tiempo y medios con los que se cuentan para realizar el proyecto, resulta imposible tratar de conseguir todos los datos de los animales de cada uno de los centros de España, estos datos no son accesibles para cualquier público, y además, el número de centros es demasiado elevado como para ir tratando cada conjunto de datos correspondiente a los animales de cada uno por separado (tal y como se hizo para los datos de las asociaciones por Comunidad Autónoma) en un tiempo tan reducido.

La solución a este problema se encontró en la posibilidad de la generación de datos sintéticos:

“Los datos sintéticos se pueden definir como información anotada artificialmente. Se genera mediante algoritmos informáticos o simulaciones. La generación de datos sintéticos generalmente se realiza cuando los datos reales no están disponibles o deben mantenerse en privado debido a la información de identificación personal (PII) o los riesgos de cumplimiento.” [5]

(Turing, s.f.)

Existen dos clasificaciones generales de datos sintéticos, los que son completamente nuevos y los que, por lo contrario, son generados a partir de una muestra de datos reales. Para el proyecto a realizar, interesa que los algoritmos de Inteligencia Artificial entrenados sean lo más fiables posibles, por ello a partir de un conjunto de datos reales, se generará una población ficticia con distribución lo más similar posible a la muestra.

Para el caso de los datos de animales, se necesitarían dos conjuntos de datos para cada centro de protección animal, uno en el que se reflejasen los animales en estado de adopción en cada uno de los centros, y otro en el que figurasen los animales adoptados. Cada conjunto de datos sería utilizado para entrenar uno de los modelos: El primero, los animales en adopción, para la funcionalidad de relacionar animales con adoptantes; y el segundo, los animales ya adoptados, para la funcionalidad de recomendar para cada nuevo animal introducido en el sistema por el usuario, un listado de centros en los que, según las adopciones pasadas, será más probablemente adoptado.

Para la consecución de la muestra de datos de animales adoptados, se han mandado correos electrónicos a un total de más de cien protectoras, de las cuales se obtuvo la vía de contacto

gracias a tener los datos de los centros almacenados correctamente como se explica en el apartado anterior, pero por diversos problemas relacionados con la protección de datos, las recientes elecciones y el hecho de que la mayoría de las asociaciones contactadas no tenían los medios para digitalizar sus registros por lo que constan en papel; solamente ha sido posible recopilar los datos de animales adoptados de la *Asociación Protectora de Animales en La Rioja* [6], que ha facilitado su base de datos tras haber mantenido una extensa conversación telefónica con su principal responsable (de profesión Ingeniero Informático) posteriormente al contacto vía email, a quien se le ha explicado minuciosamente los fines de los datos, así como el proyecto a realizar.

Siguiendo las sugerencias de la persona mencionada en el anterior párrafo, se ha contactado también con la *Fundación Affinity Petcare* [7], entidad que realiza estudios anuales relacionados con el abandono y las adopciones de animales, que podría tener una base de datos como la que se necesitaba en el proyecto, pero no ha sido posible obtener respuesta por parte de la fundación.

Por otro lado, los datos correspondientes a los animales en adopción para el algoritmo que relaciona usuarios con animales fueron obtenidos utilizando la técnica de *Web Scraping* sobre la página web de la protectora de Málaga (elegida por recomendación de nuevo, del contacto telefónico con la persona mencionada anteriormente), un proceso complejo realizado en Python mediante la herramienta *Selenium* y sin tener conocimiento previo de ella.

Como parte del *Scraper*, se han recopilado datos correspondientes a las descripciones en formato texto de los animales, con las que posteriormente y utilizando diferentes técnicas que serán explicadas en mayor detalle en apartados posteriores de la memoria correspondientes al desarrollo, se han obtenido nuevos atributos para cada animal que servirán de datos de entrada al algoritmo mencionado.

**El tamaño muestral en ambos conjuntos de datos (animales adoptados y en adopción), puede afectar gravemente a la veracidad y fiabilidad de la población ficticia generada, del mismo modo los resultados de los algoritmos no serán tan precisos, pero de nuevo y debido a la escasez de tiempo, simplemente se ha realizado el proceso para demostrar que este funcionaría correctamente en pasos futuros con datos reales, replicándolo de manera muy similar para cada una de las asociaciones.*

2.4 Tecnologías utilizadas

2.4.1 Obtención, limpieza y tratamiento de datos

- Conjunto de utilidades *Anaconda*
- Interfaz web de código abierto *Jupyter*
- Lenguaje de programación *Python*
- Librería *Pandas* para el análisis de datos

- Web Scraping¹
- *Selenium* y librerías especificadas en el desarrollo del proyecto
- Librerías de *geocodificación*
- Librerías de *fake data*
- Conversor de PDF a Excel [16]

2.4.2 Generación de datos sintéticos

- Librería de procesamiento de lenguaje natural *TextBlob*
- Análisis de sentimiento
- Preprocesamiento de texto
- Técnicas de extracción de palabras clave *TF-IDF*

2.4.3 Algoritmos de Inteligencia Artificial

- Librería *Scikit-learn* para aprendizaje automático.
- Reducción de dimensionalidad por PCA.
- Algoritmo de clustering
- Algoritmo de filtrado colaborativo

2.4.4 Desarrollo del Backend

- *FastApi* de *Python*
- Terminal de comandos

2.4.5 Desarrollo del Frontend

- Entorno de programación *Android Studio*
- *Framework Flutter*
- Lenguaje de programación *Dart*
- *Firebase Authentication*

Para la creación de la parte del Frontend de la aplicación, ha sido necesaria la instalación de *Android Studio* [8], *Flutter* y *Dart*, que se ha realizado siguiendo paso a paso el tutorial de la documentación oficial de *Flutter* [12]. Este tutorial incluye el despliegue completo del entorno de programación y su correspondiente *framework*² utilizado, además de las instrucciones para

¹ “Web Scraping es un método automático para obtener grandes cantidades de datos de sitios web. La mayoría de estos datos son datos no estructurados en formato HTML que luego se convierten en datos estructurados en una hoja de cálculo o una base de datos para que puedan usarse en varias aplicaciones.” (Ciberseguridad, s.f.) [17]

² “Un framework es un esquema o marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, una especie de plantilla que sirve como punto de partida para la organización y desarrollo de software.” (Edix, 2022) [9]

la instalación del lenguaje de programación propio de dicho *framework* y su integración en el entorno mediante la instalación del *plugin*³ y el *SDK*⁴ adecuados.

2.4.6 Otras

- *GitHub* para el control de versiones
- Dibujos de Google para la elaboración de Diagramas

³ “Los *plugins* son pequeños programas complementarios que amplían las funciones de aplicaciones web y programas de escritorio.” (Digital Guide IONOS, 2020) [13]

⁴ “Un kit de desarrollo de software (SDK) [...] incluye todos los elementos que un desarrollador podría necesitar al momento de crear aplicaciones nuevas para el producto específico y su ecosistema.” (RedHat, 2020) [14]

Capítulo 3. DESARROLLO DEL PROYECTO

3.1 Planificación del proyecto

La planificación del proyecto se ha elaborado de cara a los cinco meses de duración de este. Para esta misma, se ha dividido el proyecto en fases, y estas mismas han sido subdivididas en tareas. Además, antes de comenzar con el desarrollo del proyecto, se ha realizado un Diagrama de Gantt con la finalidad de realizar un seguimiento del progreso de las tareas mencionadas a lo largo del tiempo.

A continuación, la enumeración de cada una de las fases con una breve descripción de las tareas incluidas en cada una de ellas.

3.1.1 Formación en el ámbito en el que se desarrolla el proyecto – FASE 1

(1a) Documentación sobre aspectos básicos mediante contactos y entrevistas a Centros de Protección Animal reales, y aspectos más avanzados con la finalidad de concretar el alcance y las limitaciones del proyecto, también para matizar la utilidad de las funcionalidades del mismo de cara a optimizarlas.

(1b) Decisión de las variables más adecuadas para considerar en el conjunto de datos, pensando en su posterior visualización en la aplicación y entrada a los algoritmos que la conforman.

(1c) Investigación preliminar sobre algoritmos y herramientas que pudieran estar relacionadas o bien ser útiles para el proyecto.

(1d) Instalación de los entornos, programas y lenguajes utilizados.

3.1.2 Obtención, limpieza y tratamiento de datos – FASE 2

(2a) Búsqueda de fuentes de datos reales, conformación de la base de datos de los Centros de Protección animal españoles y recopilación de las muestras de animales a partir de las cuales se generan los datos sintéticos con los que se trabaja en el proyecto.

(2b) Tratamiento de datos.

(2c) Realización de otras operaciones sobre los datos.

(2d) Generación de datos sintéticos siguiendo la distribución de los muestrales.

(2e) Investigación sobre bases de datos compatibles con las demás tecnologías de cara a ser visualizados en la aplicación e introducidos a los modelos.

3.1.3 Algoritmos de Inteligencia Artificial – FASE 3

- (3a) Documentación e información sobre diferentes modelos de Inteligencia Artificial.
- (3b) Programación del primer modelo para la recomendación de centros.
- (3c) Programación del segundo modelo para la relación de usuarios con animales.
- (3d) Entrenamiento de los modelos con los datos utilizados en el proyecto.

3.1.4 Desarrollo del Backend – FASE 4

- (4a) Información sobre funcionamiento del Backend.
- (4b) Documentación sobre la tecnología FastAPI de Python.
- (4c) Implementación del Backend.

3.1.5 Desarrollo del Frontend – FASE 5

- (5a) Diseño del Frontend con Canva.
- (5b) Formación en el lenguaje de programación Dart.
- (5c) Inicio del Frontend con login y registro de usuarios (primera versión)
- (5d) Integración de base de datos en la aplicación
- (5e) Integración de modelos en la aplicación (“producto mínimo viable”)

3.1.6 Documentación y presentación – FASE 6

- (6a) Preparación de la documentación final (Informe del proyecto)
- (6b) Levantamiento de la aplicación final desde un dispositivo móvil real y/o desde un navegador.
- (6c) Preparación de la presentación final.
- (6d) Revisión del proyecto.
- (6e) Preparación de la presentación oral.

Las diferentes fases y tareas del proyecto han sido delimitadas temporalmente en base a las diferentes fechas destacadas señaladas en la documentación facilitada por el departamento del Trabajo de Fin de Grado, marcadas en rojo en el diagrama:

- (1*) Inicio del Trabajo de Fin de Grado (viernes 17 de febrero)
- (2*) Avance 1 del Trabajo de Fin de Grado (viernes 31 de marzo)
- (3*) Avance 2 del Trabajo de Fin de Grado (viernes 28 de abril)
- (4*) Entrega versión 1 del Trabajo de Fin de Grado (viernes 12 de mayo)
- (5*) Entrega versión definitiva del Trabajo de Fin de Grado (viernes 26 de mayo)
- (6*) Corrección de errores (viernes 2 de junio)
- (7*) Defensa del Trabajo de Fin de Grado (semana del 19 al 23 de junio)

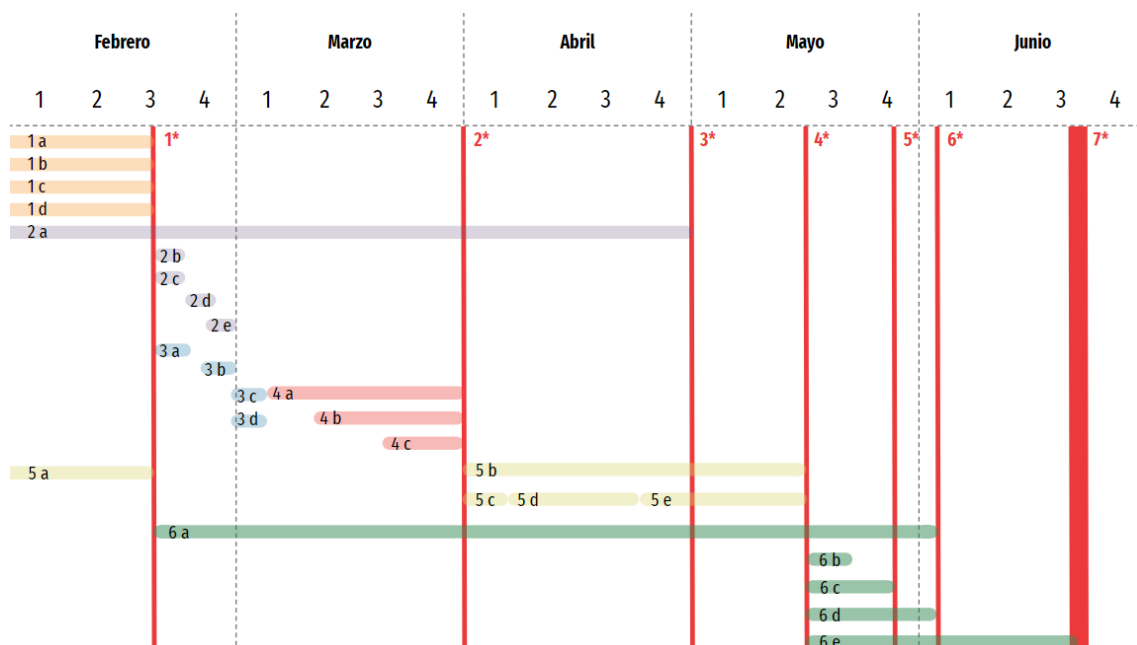


Figura 1: Diagrama de Gantt sobre la planificación del proyecto

3.2 Presupuesto

A continuación, la evaluación económica total del proyecto:

Tipo de coste	Valor	Comentarios
RECURSOS HUMANOS		
Horas de desarrollo del proyecto	9600 € (20 €/hora)	Desde febrero hasta mayo de 2023, una media de 120

		horas mensuales: 480 horas totales.
RECURSOS MATERIALES		
Ordenador portátil	1500 €	Recurso ya adquirido. Precio de mercado actual.
Ipad 8th generación	480 €	Recurso ya adquirido. Precio de mercado actual.
Apple Pencil	99 €	Recurso ya adquirido. Precio de mercado actual.
SOFTWARE UTILIZADO		
Entornos y programas	0 €	Todos los entornos y programas utilizados han sido de código libre
FUENTES EXTERNAS		
Estudios, informes...	0 €	Todas las fuentes externas empleadas para realizar el proyecto fueron gratuitas.

Tabla 8: *Presupuesto del proyecto*

3.3 Desarrollo del proyecto

En este apartado se entrará en detalle al respecto de cada uno de los puntos mencionados en el apartado anterior Planificación del Proyecto.

3.3.1 Formación en el ámbito en el que se desarrolla el proyecto – FASE 1

Los primeros pasos dados en el proyecto constituyeron las bases de la idea finalmente llevada a cabo. Para ello, se realizó un proceso exhaustivo de documentación mediante contactos con Centros de Protección Animal reales a cerca de lo que desembocó en la definición de las limitaciones y el alcance del proyecto.

Se consideró de gran importancia previa puesta en marcha del proyecto, el hecho de asegurar que la aplicación podría cubrir una necesidad real más allá de que los datos lo fueran o no, por ello durante la primera fase del proyecto además de la formación en el área de desarrollo de la aplicación (Centros de Protección Animal, abandono, procesos de acogida y adopción etc.), se realizaron diferentes encuestas telefónicas a expertos en base a la utilidad que podría tener Adoption Movement en el ámbito en el que se enfoca.

Una vez aclarado tanto el alcance como las limitaciones del proyecto, los siguientes pasos se basaron en investigar sobre posibles soluciones técnicas que pudieran contribuir a trasladar a la realidad la idea del proyecto.

En primer lugar y teniendo claro bajo la rúbrica del Trabajo de Fin de Grado de Ingeniería Matemática Aplicada al Análisis de Datos, que el proyecto debería incluir uno o varios algoritmos de Inteligencia Artificial, se decidió el desarrollo de una aplicación móvil para el sistema operativo Android, que visualizase de la manera más atractiva posible tanto la entrada como la

salida de datos proporcionados por el usuario y transformados por los algoritmos respectivamente.

Tras haber dedicado parte de la asignatura “Aplicaciones y Tendencias”, de cuarto curso del grado bajo voluntad propia al desarrollo de una aplicación móvil para Android, se decidió aprovechar el conocimiento adquirido para plasmar los resultados del Trabajo de Fin de Grado.

El entorno seleccionado para el desarrollo de la aplicación fue *Android Studio* [8], la razón de dicha elección fue la previa familiarización con el entorno y su correspondiente *framework Flutter* [10], y lenguaje de programación propio *Dart* [11], además de la motivación aportada por la completa y relativamente intuitiva documentación publicada sobre cualquiera de las tres tecnologías mencionadas.

El lenguaje de programación empleado para el desarrollo de prácticamente el resto del proyecto ha sido *Python*, utilizando la interfaz web de código abierto *Jupyter* mediante el navegador *Anaconda*. No ha sido necesario instalar durante el tiempo de desarrollo del proyecto ninguna de las tecnologías mencionadas en este párrafo, debido a que fueron las que se vinieron utilizando durante todos los cursos del grado Ingeniería Matemática Aplicada al Análisis de Datos, motivo principal de la elección.

En esta fase, a pesar de no tener claro al completo los algoritmos a utilizar en el proyecto, sí se sabía que la programación de los mismos iba a realizarse haciendo uso de nuevo del lenguaje de programación *Python*, con la intención de poner en práctica lo aprendido durante la asignatura de segundo curso del grado “Inteligencia Artificial”.

3.3.2 Obtención, limpieza y tratamiento de datos – FASE 2

Debido a las dificultades presentadas para la búsqueda de datos reales y la persistencia en este proceso, la obtención de datos ha resultado la tarea más extensa y duradera del proyecto junto con la elaboración del presente informe.

Como se explicó anteriormente en el apartado correspondiente los datos utilizados, estos se dividen en dos conjuntos: Centros y Animales, este último a su vez dividido en dos subconjuntos; Adoptados y en Adopción. Por este motivo, se explicarán los procedimientos realizados en cada caso según el conjunto en cuestión:

3.3.2.1 Centros de Animales

El proceso seguido para la obtención, limpieza y tratamiento de los datos se identifica a la perfección con lo que se conoce como proceso ETL (Extracción, Transformación y Carga) [22]. A continuación, consta el desarrollo de cada una de las fases del ETL:

- Extracción.

Al no existir una base de datos completa y unificada con información sobre todos los Centros de Protección Animal de España, y siendo este uno de los principales objetivos del proyecto

realizado, los datos crudos o *Raw Data*⁵ han sido obtenidos por separado, de los listados públicos de asociaciones correspondientes a cada una de las 17 Comunidades Autónomas españolas a excepción de Galicia y Extremadura, en cuyos casos no se ha podido obtener por motivos mencionados anteriormente.

En algunos casos, las páginas de las que se obtuvieron los listados facilitaban la opción de exportarlos directamente en diferentes formatos como CSV o Excel, siendo estos los formatos buscados para la fácil posterior importación y modificación de los datos en *Python*, y en otros casos en formato PDF, los cuales han sido transformados utilizando un conversor online de PDF a Excel [16].

En los casos en los que el sitio web no facilitaba la opción de exportar en el formato que fuera, se ha procedido a utilizar la técnica de Web Scraping [17] para la extracción de los datos mediante un programa diferente para cada página web con *Python*, haciendo uso del entorno de pruebas *Selenium*.

Partiendo del conocimiento nulo a cerca de *Selenium*, se ha realizado una amplia formación sobre dicho entorno de pruebas mediante el manual “Selenium with Python” [18] junto con la documentación oficial de la herramienta [19]. El principal problema a la hora de programar en este entorno ha sido el manejo de excepciones, con el que no estaba familiarizada, pero gracias a la descripción detallada de las mismas en el manual, todos los programas propuestos han sido desarrollados exitosamente junto con la instalación y posterior importación de los siguientes módulos y librerías de *Python* y *Selenium*, y su correspondiente documentación oficial [21] [19]:

LIBRERÍA	MÓDULOS	USO
Selenium	Webdriver	<p>Automatizar la interacción con los navegadores web, proporciona una interfaz para controlar un navegador web desde el código <i>Python</i> y permite realizar acciones como hacer clic en botones, rellenar formularios, navegar por diferentes páginas web, obtener el contenido de una página web y realizar pruebas automatizadas en una aplicación web.</p> <p>Como en este caso el navegador utilizado ha sido Google Chrome, ha sido necesario instalar el controlador <i>ChromeDriver</i> [20] que permite a <i>Selenium</i> interactuar con el navegador.</p>

⁵ “Raw Data es todo tipo de data que no ha sido procesada aún.” (carlosbermudez, 2011) [15]

Time		<p>Agregar pausas en el código con el fin de esperar a que se cargue algún elemento en una página web. No es la mejor práctica, tratándose de su utilización en el entorno de pruebas <i>Selenium</i>, ya que los métodos de esta librería agregan un tiempo fijo de espera en el código, lo cual no es eficiente, ya que, si el elemento aparece antes del tiempo de espera programado, la pausa aún se aplicará y, si tarda más tiempo en aparecer. Aun así, ha sido utilizados únicamente en casos en los que no funcionaba el método <i>WebDriverWait</i>, explicado posteriormente, un método más adaptado a la eficacia del entorno de pruebas.</p>
<i>selenium.common.exceptions</i>	<i>NoSuchElementException</i>	<p>Manejar errores cuando se intenta encontrar un elemento web que no existe en la página actual mediante bloques de código diseñados para gestionar excepciones durante la ejecución del programa.</p>
	<i>StaleElementReferenceException</i>	<p>Manejar errores cuando se intenta encontrar un elemento web que <u>ya</u> no está presente en la página actual.</p> <p>Cuando se interactúa con un elemento web utilizando <i>Selenium</i>, es posible que la página web cambie dinámicamente y que el elemento ya no se encuentre presente en el DOM (modelo de objetos del documento) de la página web. Del mismo modo que el módulo anterior, se utilizó mediante bloques de código que evitan la interrupción de la ejecución del programa.</p>
<i>selenium.webdriver.common.by</i>	By	<p>Colección de estrategias de búsqueda para localizar elementos web en una página.</p>
<i>selenium.webdriver.support</i>	expected_conditions	<p>Colección de condiciones que se utilizan para esperar a que ocurra un determinado evento en una página web antes de continuar con la ejecución del programa, se ha utilizado junto</p>

		con <i>WebDriverWait</i> para determinar la continuación de la ejecución del programa en caso de que se cumpla cierta condición, en caso contrario esperar.
<i>selenium.webdriver.support.ui</i>	WebDriverWait	Agregar pausas en el código con la finalidad de que se cargue algún elemento en una página web, pero de una manera más eficiente que la librería <i>time</i> ; pues permite definir un tiempo de espera máximo, pero continúa comprobando periódicamente si el elemento ha aparecido hasta que el tiempo de espera se agote o el elemento aparezca, lo que ocurra primero. También ofrece la posibilidad de definir condiciones personalizadas para la espera, de tipo “espera hasta que aparezca cierto elemento”.
	Select	Interactuar con formularios y listas desplegables eligiendo las opciones deseadas antes de pulsar el botón que lleve al listado de asociaciones.

Tabla 1: Librerías utilizadas para el proceso de Web Scraping

Las fuentes de las que ha partido la etapa de Extracción del proceso ETL, es decir, las páginas en las que consta el listado de asociaciones para cada Comunidad Autónoma española, constan en la bibliografía de la memoria.

- Transformación.

La salida de la etapa de Extracción fueron los conjuntos de datos en crudo correspondientes al listado de asociaciones de todas las Comunidades Autónomas Españolas.

Antes de la última fase del ETL, la carga de los datos, se aplicaron diferentes transformaciones para mejorar la calidad de estos mediante *scripts*⁶ de *Python* siguiendo una lógica dividida en diferentes pasos que se describen a continuación:

En primer lugar, los datos fueron importados en estructura *DataFrame*⁷ de la librería *Pandas* de *Python*, una de las librerías más populares para la manipulación y el análisis de datos.

⁶ “El script es un término usado en programación para hablar de los fragmentos de código usados para dar forma a herramientas” (Arimetrics, s.f.) [23]

⁷ “Conjunto de datos estructurado en forma de tabla donde [...] todos los datos de una misma columna son del mismo tipo, y las filas son registros que pueden contener datos de distintos tipos.” (Aprende con Alf, s.f.) [24]

Esta librería mediante sus diferentes métodos permite la carga de datos y transformación de los mismos en estructuras *DataFrame* a partir de formatos como Excel o CSV, los cuales se correspondían con los formatos de los datos en crudo.

Este primer paso ha resultado sencillo debida a la previa familiarización con la librería y las estructuras mencionadas, por haber sido utilizadas en repetidas ocasiones a lo largo del grado. No obstante, las posibles dudas surgidas han sido resueltas consultando la documentación oficial de la librería *Pandas* [25].

Como los datos importados estaban compuestos por los diferentes listados completos de las asociaciones de todo tipo de cada una de las Comunidades Autónomas Españolas, hubo que filtrar los datos de cada uno de manera que se pudiese desechar todo lo que no fuera una asociación de tipo Centro de Protección Animal o similar.

En este caso se pudieron diferenciar dos tipos de conjuntos a los que se aplicaron diferentes procedimientos según el caso:

- Conjuntos de datos que **sí** contaban con columnas de tipo “finalidad”, “tipo de asociación” o similar, es decir, *DataFrames* de cuyo conjunto de columnas al menos una permitiese diferenciar el ámbito de dedicación de cada asociación. Para este caso, se han filtrado todos los datos, conservando solamente los registros en los cuales este campo representase un valor relacionado con el área de estudio como por ejemplo “protección de animales y plantas”, “bienestar animal” o “naturaleza” entre otros.
- Conjuntos de datos que **no** contaban con el tipo de columnas descrito anteriormente. Para este caso, mediante funciones programadas para ello, se aplicaron filtros a la columna correspondiente a los nombres de las asociaciones de tipo: “Guarda los registros en cuyo nombre aparezca alguna de las palabras de una lista indicada”, conteniendo esta lista palabras tales como “protección animal”, “gato”, “perro”, “felino” etcétera.

Tras filtrar las asociaciones para desechar aquellas que no estuvieran relacionadas con fines animales siguiendo para cada listado de cada Comunidad Autónoma el caso más conveniente de los dos descritos, las asociaciones almacenadas entonces eran únicamente de tipo animal, pero no necesariamente centros, santuarios o refugios como se buscaba.

Para evitar incluir en el conjunto de datos utilizados en el proyecto, asociaciones relacionadas con fines animales de otro tipo al buscado (por ejemplo, antitaurinas, equinas, de educación canina etc.), se aplicó un segundo filtro de nuevo mediante funciones programadas para ello, de tipo: “Elimina los registros en cuyo nombre aparezca alguna de las palabras de una lista indicada”, estando formada la lista en este caso, por palabras relacionadas con los casos que pudieran distar demasiado de lo deseado.

El output de este paso fue entonces un conjunto de datos para cada Comunidad Autónoma (cada conjunto con una diferente estructura y formado por unas variables diferentes), en los que contaban los registros de asociaciones correspondientes únicamente a Centros de

Protección animal u otras organizaciones relacionadas con la manutención y adopción de animales.

Si bien cada conjunto de datos contaba con al menos algún atributo correspondiente a la dirección de las asociaciones, y de cara a la visualización de los datos en la aplicación mediante gráficas de mapas, fue preciso obtener los datos de la latitud y longitud de cada dirección.

Esto se consiguió mediante el módulo *Nominatim* de la librería *geopy.geocode* de *Python*. *Geopy*, proporciona una serie de herramientas y métodos para trabajar con ubicaciones geográficas, y más concretamente, *Nominatim*, facilita la geocodificación de direcciones utilizando el servicio de OpenStreetMap⁸.

Para la utilización del programa de geocodificación, ha sido necesario crear una instancia de la clase *Nominatim* que fuera a ser utilizada como un geolocalizador. Después, mediante funciones programadas para ello, se aplicó el geolocalizador a la columna de las direcciones en cada conjunto de datos, de manera que la ejecución del programa devolviese dos nuevos atributos para cada conjunto de datos correspondientes a la latitud y la longitud de cada registro.

En algunos casos, cabe destacar que el atributo de la dirección tuvo que sufrir modificaciones por no estar lo suficientemente completo. Estas transformaciones se basaron en concatenar a los campos de dirección otros presentes en los conjuntos de datos como por ejemplo “Municipio”, “Código Postal”, “Localidad” etc.

Se utilizó la documentación oficial de la librería *Geopy* [28] además del manual en línea “Cómo realizar geocodificación con GeoPy” [29] como principales fuentes de aprendizaje para la realización de los programas desarrollados en este paso.

Los atributos de cada uno de todos los conjuntos de datos fueron adaptados de manera que todos los *DataFrames* correspondientes a cada Comunidad Autónoma terminasen teniendo la misma estructura de columnas con un número de filas o registros diferente para cada caso.

Para no perder el dato de la Comunidad Autónoma a la que perteneciera cada conjunto, se creó para cada caso, un nuevo atributo con el mismo valor para todos los registros: el nombre de la Comunidad Autónoma en cuestión.

En este mismo paso se concatenaron todos los conjuntos de datos en una sola estructura *DataFrame*, sobre la cual se realizaron las transformaciones correspondientes a los siguientes pasos.

Los métodos de contacto para cada asociación tales como el correo electrónico, el número de teléfono, el sitio web y el enlace al perfil de Facebook, se consideran importantes dentro de un caso de uso con datos reales, por ello, y a pesar de no tener estos datos, se generaron

⁸ “OpenStreetMap proporciona datos de mapas para miles de sitios web, aplicaciones móviles y dispositivos de hardware” [26]

con librerías de *fake data*⁹ con el fin de acercar el proyecto a una futura aplicación real de los datos en la medida de lo posible.

Las librerías empleadas para la generación de *fake data* en este paso fueron las siguientes:

LIBRERÍA	USO
Faker	Generar datos aleatorios realistas de una manera sencilla. Los números de teléfono y los links al perfil de Facebook fueron generados con un solo método de esta librería.
Random	Incluye diferentes funciones para generar números aleatorios, fue utilizada para aleatorizar la longitud de los nombres de usuario de los correos electrónicos y así aproximar los resultados a un caso real.
Re	Conjunto de métodos para trabajar con expresiones regulares, las cuales fueron utilizadas para limpiar los nombres de cada registro de manera eficiente y en un solo paso (eliminar caracteres especiales), y permitir que estos nombres limpios fueran empleados posteriormente para generar los correos electrónicos, los enlaces a Facebook y los sitios web de cada registro.

Tabla 2: *Librerías utilizadas para la generación de datos ficticios*

De nuevo el conocimiento adquirido para programar haciendo uso de las librerías mencionadas, provino de la documentación oficial de *Faker* [30], *Random* [31] y *Re* [32].

Además, para poder poner en práctica los métodos de la librería de expresiones regulares de *Python*, *re*, ha sido necesario adquirir ciertos conocimientos básicos sobre las mismas. Para ello se ha empleado la “Guía para entender y usar expresiones regulares” [33]

De cara a facilitar las futuras relaciones del conjunto de datos de los centros con el de los animales, surgió la idea de crear un atributo con el valor correspondiente a la categoría de cada centro siendo los posibles: “canino”, “felino” o “mixto”.

Con la finalidad de aproximar la distribución de los datos de animales asociados a los centros a la realidad, en base al nombre de los registros se aplicó una función que atribuyó en cada caso una categoría entre las mencionadas. Para la elaboración de esta función se crearon

⁹ “Fake data es aquel que no surge, no es provocado o no se obtiene de la realidad de la que procede y que, de una forma u otra, se manipula con o sin intención” (Cera, 2019) [27]

dos listas, una con palabras relacionadas con la raza felina y otra con la canina, pues los perros y los gatos son los animales en los que se enfoca el actual proyecto.

La función se programó para decidir sobre la columna de nombres, si de las palabras que formaban el de cada registro alguna se incluía en cada una de las listas descritas. En función de lo que ocurriera, se le atribuiría un valor u otro a la nueva columna creada para la clasificación; y en caso de que el nombre no se pudiera clasificar por no estar presente en él ninguna de las palabras definidas en las listas, el registro sería evaluado como “mixto”.

Cabe destacar que este proceso se realizó principalmente para aportar coherencia entre los nombres de las asociaciones y su correspondiente categoría. En apartados posteriores de la memoria se explica cómo las distribuciones de los datos en base a su categoría fueron reajustadas.

Con el último paso mencionado finalizaría la fase de Transformación del proceso ETL, siendo la siguiente y última fase la correspondiente a la carga de los datos, detallada más adelante.

Como resultado de las operaciones y transformaciones detalladas anteriormente, se obtuvo el siguiente conjunto de datos descrito resumidamente a continuación:

DATOS DE CENTROS DE PROTECCIÓN ANIMAL	
ATRIBUTO	PROCEDENCIA
Nombre	Dato real.
Direccion	Dato real.
Latitud	Datos generados a partir de la dirección.
Logitud	
CCAA	Dato añadido a cada conjunto por Comunidad Autónoma antes de fusionar todos los conjuntos en uno.
Teléfono	Datos generados aleatoriamente.
URL	
Correo	

Facebook	
Categoría	Datos generados en base a las palabras contenidas en el nombre de la asociación.

Tabla 3: *Datos de centros de protección animal*

- Carga.

La última fase del proceso ETL fue la fase de carga, enfocada en este caso en cargar los datos en estructuras *DataFrame* de *Python*.

En general, el proceso de carga se enfoca en decidir la estructura óptima en la que almacenar los datos previamente transformados para que estos puedan ser empleados posteriormente. Como las operaciones siguientes a la carga irían a ser realizadas haciendo uso principalmente de la librería *Pandas* de *Python* mencionada anteriormente, la estructura seleccionada para almacenar los datos del proyecto ha sido *DataFrame*.

Es importante destacar que, para la elección del tipo de estructuras de almacenamiento de datos para el proyecto, se ha tenido en cuenta en todo momento el tamaño de los datos, sabiendo que, de ser mayor, la decisión de este tipo de estructuras implicaría directamente limitaciones relacionadas con el rendimiento por problemas de capacidad de almacenamiento, tal y como se ha aprendido en la asignatura “Almacenamiento Masivo de Datos” de tercer curso del grado.

3.3.2.2 Datos de Animales

La obtención de datos de animales reales ha sido una tarea sin éxito como se ha explicado en anteriores apartados de la memoria; y por ello se ha tomado la decisión de generar datos sintéticos para dos muestras, una para el subconjunto de animales adoptados, y otra para el subconjunto de animales en adopción.

Las muestras, como ya se ha mencionado, han sido una de ellas proporcionada por un responsable de un centro de protección animal conmovido por el proyecto realizado (animales adoptados), y la otra extraída de una página web mediante el proceso de *Web Scraping* (animales en adopción).

La muestra de animales en adopción, extraída mediante *Web Scraping*, ha tenido que pasar por varias transformaciones hasta alcanzar el objetivo de que esta misma conformase el conjunto de atributos en su correcto estado para la posterior generación de la población sintética a partir de ella. A continuación, la explicación del proceso detallado de las transformaciones realizadas:

Siguiendo un mecanismo similar al de los *Scrapers* realizados para el caso de la obtención de los datos de centros, se han obtenido una serie de atributos, que, con su total de 207 registros, conformaron la muestra en crudo de animales en adopción.

Entre los atributos mencionados, se extrajo la descripción para cada uno de los animales; un texto compuesto por cadenas de caracteres de diferentes tipos, que en su conjunto se refieren al comportamiento, pasado y preferencias del animal entre otras cosas.

En base a las descripciones de cada animal se generaría el algoritmo de inteligencia artificial para relacionarlos con usuarios mediante un test realizado por los últimos a cerca del estilo de vida y diferentes características que definirían la compatibilidad de ambos sujetos. Para eso, ha sido importante realizar una serie de operaciones y transformaciones sobre la variable “descripción”.

En primer lugar, se han limpiado los datos eliminando aquellos registros cuya descripción fuese nula, ya que no serían útiles para el modelo y tampoco tendría sentido tratarlos de otro modo.

Lo siguiente, ha sido realizar un análisis de sentimiento sobre la columna entera mediante la librería *TextBlob* de *Python*, una librería de lenguaje natural que permite realizar tareas relacionadas con el procesamiento de texto de una manera relativamente sencilla [42].

El análisis de sentimiento implica la evaluación de la actitud, la emoción o la opinión expresada en un texto, categorizándola como positiva, negativa o neutral. Para este caso, se ha medido la polaridad del sentimiento de cada uno de los textos, los valores que puede tomar esta polaridad se comprenden entre -1 y 1, siendo los valores negativos los que simbolizan la negatividad del texto, los valores cercanos a cero la neutralidad, y los valores positivos la positividad [43].

La respuesta obtenida para cada descripción se ha almacenado en una nueva columna del conjunto de datos en cuestión que sería utilizada más adelante en pasos que se explican posteriormente en la memoria.

Para las descripciones utilizadas como entrada en la función de polarización, no han existido resultados negativos, lo cual tiene su parte de lógica debido a que los textos aportan información sobre animales teniendo como objetivo principal que estos llamen la atención de los usuarios de la página y sean adoptados, pero sí han resultado textos con sentimientos neutros. Los registros con cuyo valor de polaridad fuera igual a cero, han sido interpretados como “seminegativos” ya que bajo criterio propio se ha considerado que una parte de los mismos ha debido ser lo suficientemente negativa para que no fueran catalogados como positivos. Todos los demás valores de polaridad de los registros, mayores que cero, se han interpretado como “positivos”.

A continuación del análisis de sentimiento, se ha elaborado un preprocesamiento de datos eliminando caracteres especiales, saltos de líneas y puntuaciones en los textos haciendo uso de la librería *re* de *Python*, explicada en anteriores partes de la memoria. Todas las palabras se han convertido en minúsculas, y también se ha procedido a la eliminación de *palabras vacías*¹⁰.

La eliminación de palabras vacías mencionada ha sido realizada por medio de la librería *NKT* (*Natural Language Toolkit*) [45] de *Python*, que proporciona una lista de palabras vacías

¹⁰ “Palabras vacías es el nombre que reciben las palabras sin significado como artículos, pronombres, preposiciones, etc. que son filtradas antes o después del procesamiento de datos en lenguaje natural” (Wikipedia, 2019) [44]

comunes en varios idiomas según el que se le especifique. En este caso español. Para aprender a utilizar la librería, se ha consultado la documentación oficial de la misma.

Como parte del preprocesamiento, y haciendo uso de nuevo de la librería mencionada en el párrafo anterior y de su documentación, se ha *tokenizado* cada definición, dividiendo el texto obtenido del anterior paso en unidades más pequeñas llamadas *tokens*.

El preprocesamiento de los textos de la descripción de los animales ha sido la fase preliminar para poder proceder posteriormente a la identificación de temas. Para ello se han utilizado técnicas de extracción de palabras claves *TF-IDF* “*Term Frequency-Inverse Document Frequency*” o “*Frecuencia de Término-Frecuencia Inversa de Documento*” en español.

Las técnicas *TF-IDF* utilizan una fórmula matemática para calcular la importancia relativa de cada palabra dentro de la descripción en base a la frecuencia de aparición de la palabra en el texto y en el conjunto de textos. La frecuencia de término *TF* mide la aparición de una palabra específica, cuantas más veces aparezca una palabra en el documento mayor será esta frecuencia, y por otro lado, la frecuencia inversa del documento *IDF* mide la rareza de una palabra en el conjunto total de documentos, las palabras que aparezcan con frecuencia en muchos de los documentos tendrán una frecuencia inversa del documento baja mientras que las palabras que aparezcan con menor frecuencia tendrán una frecuencia inversa del documento alta.

La fórmula *TF-IDF* es la siguiente:

$$TF - IDF = (Frecuencia\ del\ término) * \log \left(\frac{Número\ de\ documentos}{Frecuencia\ inversa\ de\ documento} \right)$$

En la práctica de la programación, se ha utilizado un modelo vectorizador *TF-IDF* [46], que en primer lugar se ajustó a la columna de descripción ya que fue la que conformaba el total de los textos a analizar, en segundo lugar, transformó estos textos en una matriz numérica donde cada fila representó una descripción diferente y cada columna una palabra, y por último seleccionó los nombres de las palabras claves a partir de las columnas de la matriz almacenándolas en una lista.

El objetivo de esta técnica fue obtener una estructura *DataFrame* con una columna para cada palabra clave, y en las filas su valor correspondiente para cada registro de animales.

Para realizar este proceso, además de las documentaciones oficiales de las librerías y herramientas utilizadas, se ha consultado el Manual “Understanding TF-IDF with Python example” de Sefidian Academy [47].

Las columnas obtenidas de la técnica explicada anteriormente correspondientes a cada palabra relevante fueron demasiadas y en algunos casos redundantes, por lo que, como solución a esto se realizó un intento fallido de agrupar las palabras por similitud mediante un algoritmo de *K-means*¹¹ de Inteligencia Artificial. El problema de este modelo fue la escasez de palabras para el

¹¹ “K-means es un algoritmo de clasificación no supervisada (clusterización) que agrupa objetos en k grupos basándose en sus características. El agrupamiento se realiza minimizando la suma de distancias

entrenamiento del mismo que se reflejó en unas agrupaciones poco claras y sin sentido, pero a continuación se explican los pasos realizados durante el intento.

En primer lugar, se utilizó el modelo *Word2Vec* [49], que emplea redes neuronales para aprender representaciones vectoriales de palabras a partir de grandes conjuntos de datos de texto. Estas representaciones vectoriales encontrarían similitudes semánticas y sintácticas entre palabras para posteriormente agruparlas utilizando el modelo de *clustering K-means* [50] en base a las características de la muestra.

Después de unos primeros resultados del modelo de *clustering* explicado anteriormente, se realizó la “regla del codo” [51] interpretándola mediante una gráfica para averiguar el número óptimo de clusters para el algoritmo. Este método se basa en la idea de que a medida que se aumenta el número de clusters, la varianza intra-cluster o la medida de la dispersión de los puntos dentro de cada clúster disminuye, ya que los clusters tienden a ser más específicos y pequeños. No obstante, a partir de cierto punto el beneficio de añadir más clusters comienza a disminuir y la varianza intra-cluster se estabiliza. Este punto representa el número óptimo de clusters para el algoritmo, que en este caso ha resultado ser aproximadamente 7.

Como consecuencia de que este procedimiento no funcionase como se esperaba debido al escaso tamaño muestral, como ya se ha comentado antes, se ha procedido a realizar la agrupación de palabras por similitud manualmente.

Para la agrupación se definió una función para que, dada una lista con las categorías agrupadas, se crease en el *DataFrame* una nueva con la agregación de los valores de las contenidas en la lista. Para eliminar las distorsiones creadas por la agregación, se ajustaron los porcentajes de manera que todas las categorías sumasen un total de 1 para cada registro con sus correspondientes equivalencias “normalizando” su distribución.

Por último, se seleccionaron aquellas columnas o categorías para las que no estaba claro el sentimiento de la palabra por sí misma, y se le añadió una polaridad en base a la calculada anteriormente para el texto de procedencia, indicando con valores negativizados aquellos catalogados con polaridad “seminegativa”, y dejando como estaban (en positivo), los que se correspondían con una polaridad “positiva”.

Una vez realizadas todas las transformaciones y operaciones descritas anteriormente sobre el conjunto de datos, se procedió a la generación de los datos sintéticos en base a las distribuciones de la muestra para que estos tuvieran la mayor coherencia posible, tomando conciencia en todo momento, que la mayor cantidad de muestras o incluso la existencia de una muestra para cada subconjunto por Comunidad Autónoma, hubiera concluido en una población con resultados mucho más precisos que los obtenidos con una única fuente de datos tomada como muestra para cada caso.

El proceso de la generación de datos sintéticos, fue bastante similar para el subconjunto de datos de animales adoptados y el subconjunto de datos de animales en adopción, en ambos

entre cada objeto y el centroide de su grupo o clúster. Se suele usar la distancia cuadrática.” (Unioviado, s.f.) [48]

casos, a partir de cada una de las muestras por separado, se obtuvo una población de un total de medio millón de datos en base a las frecuencias de cada valor único que pudiera tomar la variable en la muestra y en casos en los que la muestra no proporcionara los atributos necesarios, en base a un informe elaborado por la *Fundación Affinity* el pasado año 2022: “Infografía Él nunca lo haría. Estudio de abandono y adopción 2022” [52].

Las librerías utilizadas para la generación de datos sintéticos de manera aleatoria y en base a una distribución dada, han sido *Random*, utilizada en anteriores ocasiones y por tanto ya explicada en anteriores apartados de la memoria, y *Numpy* [53], una librería de *Python* para la computación científica que proporciona herramientas para la manipulación de matrices y *arrays*¹² multidimensionales.

Como bien se explica anteriormente, no todas las frecuencias de los valores únicos proceden de la muestra, además, existen atributos que no tendría sentido genera a partir de la muestra como por ejemplo el nombre del animal, por ello en las dos siguientes tablas se especifica la procedencia de cada atributo en cada uno de los subconjuntos de datos de animales:

POBLACIÓN DE DATOS DE ANIMALES EN ADOPCIÓN	
ATRIBUTO	PROCEDENCIA
Nombre	Generado aleatoriamente en base al sexo del animal utilizando <i>Mimesis</i> [55] de <i>Python</i> , una librería de generación de datos sintéticos que permite generar datos realistas de manera aleatoria. Anteriormente, esta librería contenía métodos para generar nombres de animales, pero en las nuevas actualizaciones de la misma, estos métodos no constan. Entonces se han generado los nombres en base a métodos que devuelven el primer nombre de personas ficticias, especificando el idioma español.
id_animal	Creado a partir de una función que asocia un índice para cada registro del 1 al 500000 basándose en la cantidad de registros existentes en el conjunto de datos.
Sexo	Dos posibles valores categóricos: “Macho” y “Hembra”. Extrapolados a la población tras sacar las frecuencias de cada valor único en la muestra y pasarlas como parámetros en la función de generación aleatoria de datos.

¹² “El módulo array de la librería estándar de Python permite declarar un objeto que es similar a una lista pero sólo puede almacenar datos del mismo tipo.” (Jiménez, 2019) [54]

Raza	28 posibles valores categóricos, 26 para la especie canina y 2 para la felina, estos últimos unificados en un solo valor “gato”. Generados a partir de la distribución de los valores en la muestra como en el caso anterior.
Especie	Determinada a partir de una función que evalúa los valores del atributo “Raza”, devolviendo el valor “Felina” para aquellos registros en cuya raza constase “Gato”, y devolviendo “Canina” en cualquier otro caso.
Tamaño	<p>Determinado en función de los valores del atributo “Especie”, para los registros en cuya especie figurase “Felina”, los valores de tamaño tomarían en todos los casos “Pequeño”, y para los registros en cuya especie figurase “Canina”, los valores del tamaño fueron rellenados en base a las frecuencias de la muestra filtrada por los registros de perros de manera idéntica a como se hizo para las variables “Sexo” o “Raza” por ejemplo.</p> <p>Los valores posibles para este atributo fueron: “Grande”, “Mediano”, “Muy Grande” o “Pequeño”, y es importante mencionar que estos tamaños hacen referencia al futuro tamaño del animal, sin tener en cuenta la edad del mismo.</p>
Edad	15 posibles valores categóricos, desde “Menos de 1 año” hasta “14 años”. Generados a partir de la distribución de los valores en la muestra y convertidos en un formato perfectamente entendible como el que se ha mencionado.
Categorías (42)	42 atributos correspondientes a las categorías obtenidas de las operaciones realizadas sobre las descripciones de la muestra, explicadas en detalle anteriormente. Generadas y posteriormente adaptadas mediante una función que equilibra las proporciones de cada registro en base a las anteriores, pero de manera que el total de las categorías sumen 1, representando cada puntuación un porcentaje de un 100% total.

Tabla 4: *Datos de animales en estado de adopción*

El aprendizaje de la generación de datos sintéticos a partir de una muestra fue completamente autodidacta, y por ello para comprobar que las funciones programadas relacionadas con la generación de la población en base a los valores y frecuencias de la muestra funcionasen

correctamente, se elaboraron gráficas para algunos atributos como las que constan a continuación:

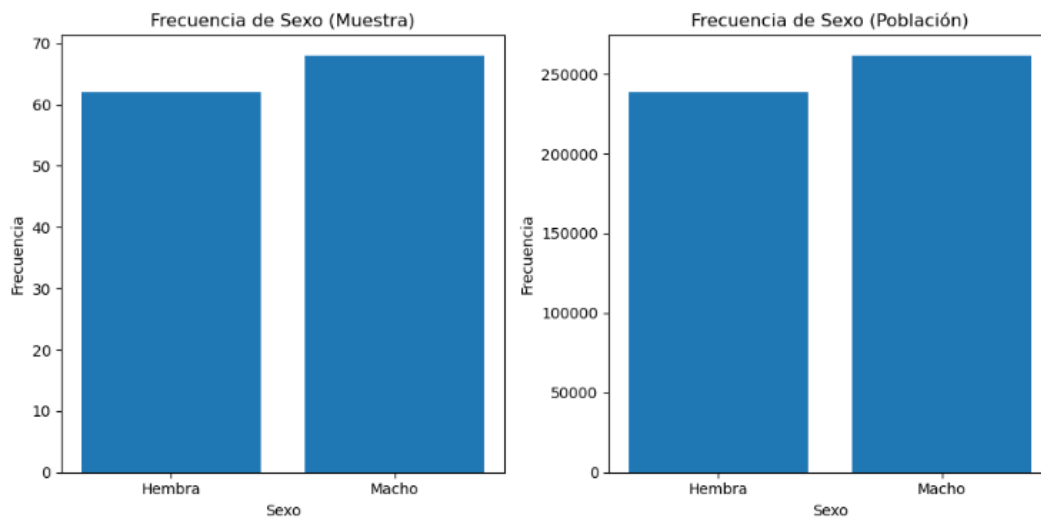


Figura 2: Comparación de frecuencias de muestra y población sintética

Gráficos de barras de frecuencias de la muestra y de la población, a simple vista parecen idénticos, y como las escalas de los datos difieren tanto, sería inútil intentar representar ambos gráficos en un mismo eje. La solución tomada para esto fue la elaboración de otro gráfico de barras, donde los ejes se correspondiesen con las proporciones y no con los valores reales. Se rebajó la opacidad de los colores de las representaciones de ambas muestras para ver las partes superpuestas de las mismas con claridad.

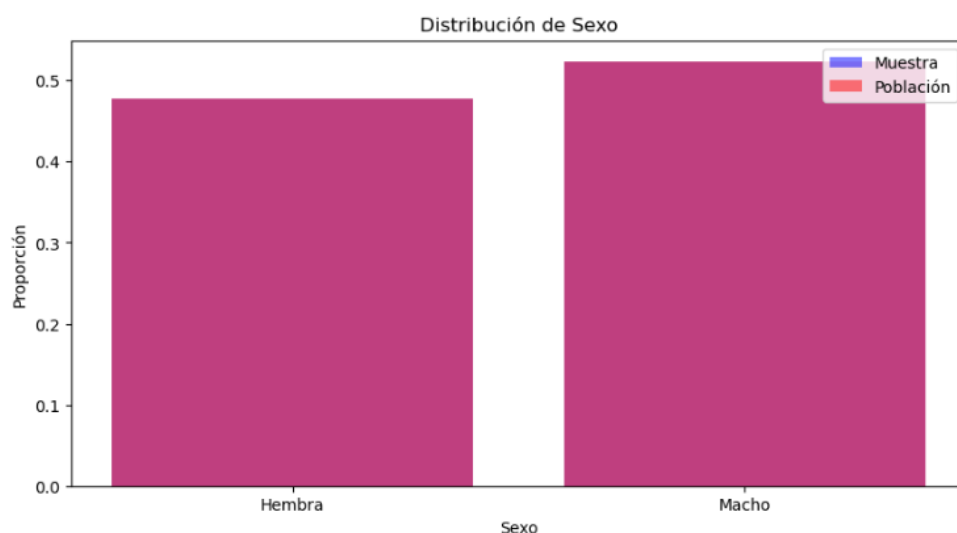


Figura 3: Superposición de frecuencias de muestra y población sintética

Efectivamente las frecuencias de la población son exactamente idénticas que las de la muestra a partir de la cual los datos fueron generados, tras lo cual se concluyó, que las funciones de generación de datos sintéticos programadas funcionaron correctamente y como se esperaba.

Para el caso de la muestra de datos de animales adoptados, las operaciones realizadas han sido mucho más sencillas y no se ha realizado ninguna que no haya sido explicada en anteriores partes de la memoria.

POBLACIÓN DE DATOS DE ANIMALES ADOPTADOS	
ATRIBUTO	PROCEDENCIA
id_animal	Generado exactamente del mismo modo que para el subconjunto de datos de los animales en adopción.
Sexo	No consta como variable en la muestra, y tampoco en el informe de la <i>Fundación Affinity</i> “Infografía Él nunca lo haría. Estudio de abandono y adopción 2022”. Por ello, las frecuencias y valores utilizadas para determinar la distribución del sexo en la población fueron las mismas que tuvieron la anterior muestra y población: Animales en adopción.
Raza	Generada del mismo modo que para el subconjunto de datos de los animales en adopción, pero esta vez en base a los valores de la muestra de animales adoptados, pudiendo tomar un total de 20 valores categóricos diferentes, 19 para perros y 1 para gatos.
Especie	Determinada de nuevo a partir de una función que evalúa la raza al igual exactamente que en el conjunto de datos de animales en adopción. Toma dos posibles valores categóricos: “Canina” y “Felina”.
Tamaño	Generado del mismo modo que para el conjunto de datos de animales en adopción en base a las especies, de manera que todos los registros en cuya especie constase el valor “Felina” tuviesen tamaño “Pequeño”, y en caso contrario se les atribuyese un valor en base a los de la muestra y sus frecuencias, pudiendo ser estos valores: “Pequeño”, “Mediano” y “Grande”.
Microchip	Las proporciones de este atributo fueron extraídas del informe realizado por la <i>Fundación Affinity</i> “Infografía Él nunca lo haría. Estudio de abandono y adopción 2022”. Se diferenciaron valores para cada una de las especies, siendo un 89% los perros que contasen con microchip y un 51% en el caso de los gatos.

Tiempo	<p>Este atributo hace referencia a la cantidad de tiempo de permanencia del animal en el centro, en la muestra se ha obtenido a partir de la resta de otros dos: “Fecha de último estado” (siendo la muestra filtrada previamente para contar solamente con los valores cuyo último estado fuera “Adopción”) menos “Fecha de entrada”. Los valores obtenidos han estado comprendidos en un rango desde los cero meses hasta los ocho, ya que los datos de esta muestra solamente comprendían los de animales ingresados en el último año.</p> <p>Para crear una distribución un poco más realista, se ha buscado el dato del tiempo que suele estar un animal en una protectora (media mínima y media máxima) siendo este de los 3 a los 10 meses, obtenido del informe realizado por la <i>Fundación Affinity</i> “Infografía Él nunca lo haría. Estudio de abandono y adopción 2021” [56].</p> <p>Los datos proporcionados por el estudio, bajo criterio propio y bajo lo consultado en entrevistas con miembros de diferentes asociaciones, siguió pareciendo bajo, por lo que se introdujeron <i>outliers</i> en los valores de la muestra desde los 9 meses hasta los 3 años con frecuencia de 1 vez, y a partir de los valores actualizados se generó la muestra de datos sintéticos.</p>
Edad	<p>Fueron generadas aleatoriamente con el formato “x años y x meses”, teniendo en cuenta los valores del atributo “Tiempo” para cada registro; pues las edades siempre tendrán que ser mayores que el tiempo que los animales llevan en adopción, pudiendo tomar valores desde los cero meses hasta los 14 años.</p>

Tabla 5: *Datos de animales adoptados*

Tras tener los datos limpios de los conjuntos de datos de centros y animales, antes de ejecutar la última fase del ETL para ambos casos, es decir, la carga de los datos fue importante establecer relaciones entre ambos conjuntos.

Para ello se crearon nuevos atributos para cada uno de los conjuntos:

En el conjunto de centros:

- Capacidad
- Ocupación
- id_centros

En el conjunto de animales en adopción y en el de animales adoptados:

- id_centro

Y se modificaron algunos de los existentes en el conjunto de centros para adecuarlos a las proporciones reales en base al estudio realizado por la *Fundación Affinity* “Infografía Él nunca lo haría. Estudio de abandono y adopción 2022”.

En primer lugar, las estadísticas proporcionadas por el informe relacionadas con las categorías de centros que se dedican a albergar animales de especie canina, felina o mixta, no coincidían con las que constaban el conjunto de datos tras haber creado la columna “Categoría” en base a las palabras contenidas en los nombres de las asociaciones.

Para visualizar las estadísticas anteriores se elaboró la siguiente gráfica:

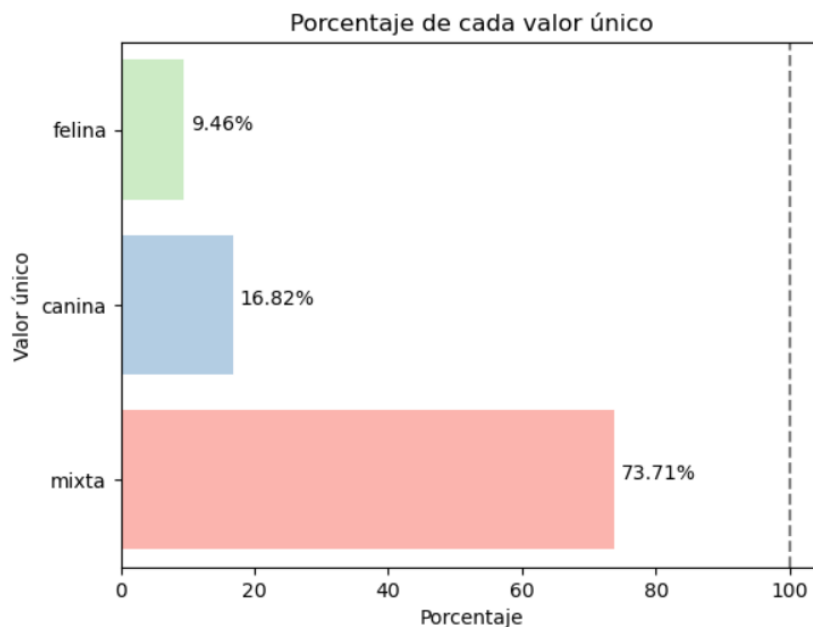


Figura 4: *Porcentajes iniciales de la distribución de una muestra*

Los nuevos porcentajes acorde con el estudio mencionado fueron los siguientes:

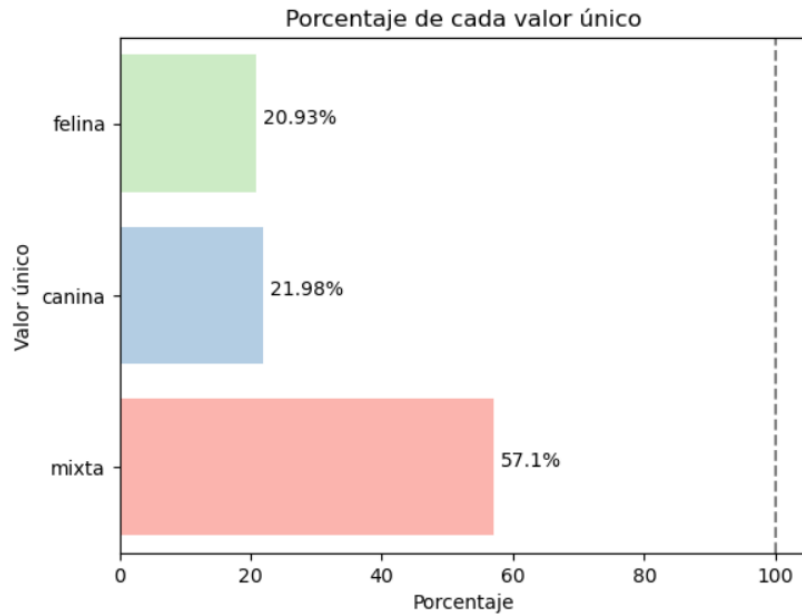


Figura 5: *Porcentajes finales de la distribución de una población sintética*

La manera en la que se ajustaron los porcentajes, para no modificar los registros que ya tenían asociados valores de categorías caninas y felinas, fue añadir valores de la categoría mixta a las restantes hasta alcanzar los porcentajes indicados.

En segundo lugar, se creó una nueva columna correspondiente a la capacidad de las protectoras en base a las estadísticas del estudio mencionado anteriormente.

Las estadísticas obtenidas, venían dadas para tres categorías de tamaño: grande, mediana y pequeña según la capacidad máxima de animales que pudiera alojar cada centro diferenciando los casos de centros entre felinos y caninos. Para realizar este paso se creó en primer lugar una columna de tamaño con la frecuencia correspondida a las estadísticas para los centros felinos y caninos, y para los centros mixtos, cuyas estadísticas no constaban en el estudio, estas tres posibles categorías se repartieron en la columna de datos de manera aleatoria.

A partir de la columna correspondiente al tamaño de los centros con sus tres posibles valores categóricos, se creó otra nueva columna correspondiente a la capacidad, para añadir valores a este atributo se creó una función que generase valores enteros de ocupación aleatorios en diferentes rangos en función del tamaño previamente asociado. Valores entre 20 y 100 para centros pequeños, entre 100 y 300 para centros medianos, y entre 300 y 500 para centros grandes.

A partir de la columna nueva correspondiente a la capacidad máxima de los centros, se creó una columna de ocupación, que se generó mediante otra función para dar valores aleatorios en función de la capacidad generada anteriormente, pudiendo tomar valores para la ocupación en cada caso en un rango desde el 25% de la capacidad máxima, hasta un 175% en el caso posible y real de que las instalaciones del centro estuvieran saturadas.

Los valores de la categoría ocupación de los centros debían sumar medio millón en total para poder en pasos posteriores asociar cada centro a un animal del conjunto de datos de animales en adopción, pero al contar con un número tan elevado de registros de animales para un número tan pequeño de centros, se observó el problema de que la ocupación en casi todos los casos iba a ser tan elevada que el resultado de esto concluiría en un conjunto de datos de centros mayoritariamente saturados, lo cual dista demasiado de la realidad.

Para solucionar este problema, se recortó el conjunto de datos de animales en adopción cogiendo una muestra aleatoria de datos de 200.000 registros de los 500.000 que había. Así se pudieron asociar los animales a los centros correctamente de manera que, en el conjunto de datos de animales, el número de animales asociados a cada centro fuera el mismo que el número de ocupación de ese centro en su correspondiente conjunto de datos.

Para el caso del conjunto de centros de animales adoptados, esto no fue un problema debido a que este conjunto representaría el registro histórico de adopciones para cada centro, entonces no se tendría que tener en cuenta la ocupación de las instalaciones de cada uno, simplemente se asociaron registros de animales a centros en proporción a la capacidad de los mismos de manera que la cantidad de animales adoptados en un centro pequeño no superase la cantidad de animales adoptados en un centro grande.

Después de establecer las relaciones de los conjuntos mediante la presencia de atributos de unos en otros, se ha realizado una última limpieza de columnas de cada conjunto, conservando solamente las necesarias para la conformación de la base de datos. El resultado de este proceso culminó en la base de datos representada en el siguiente modelo Entidad Relación¹³:

¹³ “Un modelo entidad-relación es una herramienta para el modelo de datos, la cual facilita la representación de entidades de una base de datos.” (Wikipedia, 2023) [56]

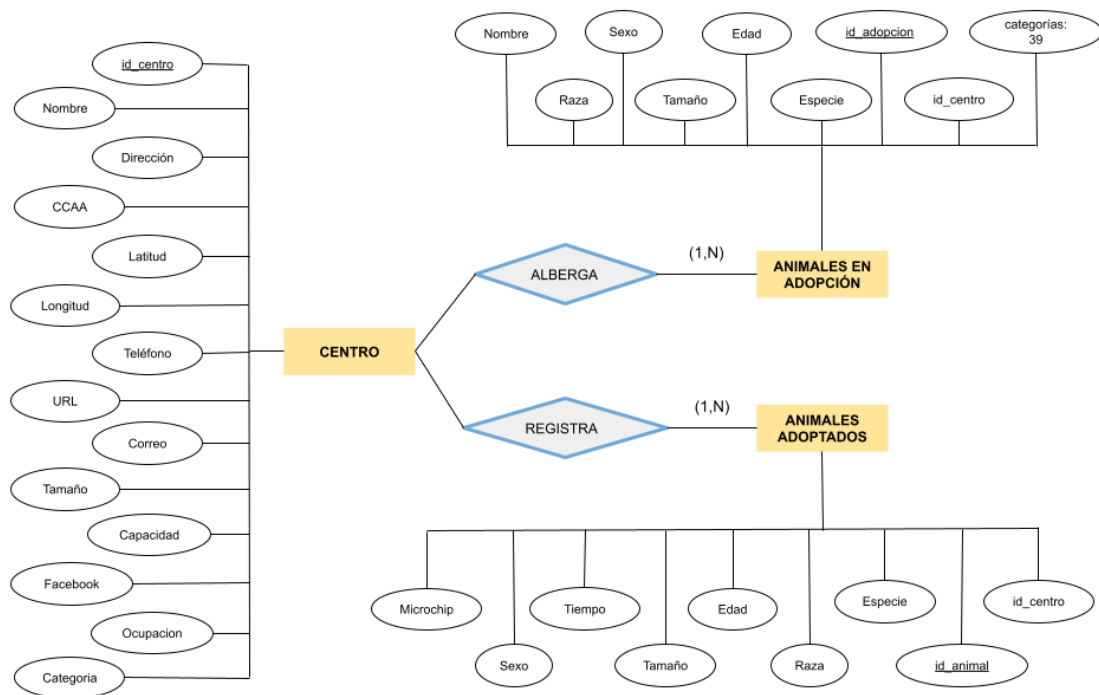


Figura 6: Diagrama ER de los conjuntos de datos

Donde la clave foránea mediante la cual se relaciona el conjunto de centros con los dos de animales es "id_centro".

**Para mejorar la visualización del modelo Entidad Relación, no se han representado las categorías correspondientes al conjunto "Animales en adopción", simplemente se han simbolizado como un solo atributo que representa las treinta y nueve siguientes categorías: "cat_abandonado", "cat_cariño", "cat_cercania", "cat_acogida", "cat_actividad", "cat_dependencia", "cat_adaptacion", "cat_docil", "cat_agradecido", "cat_agresivo", "cat_tenso", "cat_alegre", "cat_inteligencia", "cat_miedo", "cat_convivencia", "cat_correa", "cat_carretera", "cat_aventura", "cat_niños", "cat_belleza", "cat_ppp", "cat_bruto", "cat_triste", "cat_tranquilidad", "cat_desubicado", "cat_recuperacion", "cat_cronico", "cat_historial", "cat_fortaleza", "cat_independencia", "cat_maltrato", "cat_gato", "cat_perro", "cat_pelo", "cat_macho", "cat_hembra", "cat_ladra", "cat_ruido".*

3.3.3 Algoritmos de Inteligencia Artificial – FASE 3

Uno de los puntos más importantes del proyecto para el fin de grado de Ingeniería Matemática Aplicada al Análisis de Datos, es la programación de algún algoritmo de inteligencia artificial como parte del mismo.

Además de los descritos en la anterior fase para el tratamiento de los datos, en el proyecto se incluyen dos modelos elaborados desde cero para cubrir dos necesidades específicas respectivamente; un primer modelo que en base al conjunto de datos de animales adoptados recomiende un centro en el que más posibilidades tenga de ser adoptado un nuevo animal para sus características concretas, y un segundo modelo que relacione usuarios con animales del conjunto de animales en adopción en base a las características buscadas y el estilo de vida del usuario.

3.3.3.1 Algoritmo *K-Modes* para la recomendación de centros

El objetivo que se persiguió con el algoritmo a desarrollar fue obtener la salida de los centros en los que mayormente un animal pudiera ser adoptado, a partir de la entrada de las características de dicho animal. Para la realización de este algoritmo se ha utilizado el conjunto de datos de animales adoptados, un registro histórico de animales adoptados asociados cada uno a un identificador de un centro, que se corresponde con un registro del conjunto de datos de los centros.

La idea de funcionamiento se basa en agrupar animales por sus características en grupos similares, cuando se introducen los datos de un nuevo animal proporcionados por el usuario, se realiza una predicción que devuelve el grupo al que pertenece el nuevo animal según los resultados del algoritmo, y después se analiza ese mismo grupo haciendo un conteo de las asociaciones que constan en cada registro y sus frecuencias. Para los resultados recibidos por el usuario se muestra un ranking de asociaciones en base a la frecuencia de las mismas dentro del grupo de animales similares al que se asocie el nuevo registro.

El algoritmo seleccionado para el desarrollo de esta funcionalidad ha sido *K-Modes* [57], una extensión del algoritmo *K-Means* utilizado para la agrupación de datos mayoritariamente categóricos en lugar de datos numéricos.

Para entender el funcionamiento del algoritmo *K-Modes*, es necesario explicar de dónde procede, del algoritmo *K-means*, que se basa en conceptos matemáticos y estadísticos para agrupar objetos en clusters teniendo como objetivo encontrar la mejor manera de separar un conjunto de datos en K grupos, donde cada grupo contiene objetos que son más similares entre sí que con los objetos en otros grupos.

Para lograr esto, el algoritmo utiliza la distancia euclidiana entre los objetos en el espacio de características. El espacio de características es un espacio abstracto en el que cada objeto es representado por un vector de características. La distancia euclidiana es una medida que se utiliza para calcular la distancia entre dos puntos en el espacio de características.

Una vez definido el espacio de características y se ha seleccionado el valor de K, para lo cual existen diferentes técnicas empleadas en el proyecto que se explicarán más adelante, el algoritmo inicializa K centroides en el espacio de características. El centroide se utiliza para representar el centro de cada clúster y se calcula como la media de todos los objetos del clúster.

A continuación, el algoritmo asigna cada objeto en el conjunto de datos al centroide más cercano utilizando la distancia euclidiana. Cada objeto es asignado al clúster cuyo centroide está más cerca del objeto, y este proceso se repite hasta que todos los objetos hayan sido asignados a un clúster.

Una vez que se han asignado todos los objetos a un clúster, el algoritmo recalcula los centroides de cada clúster como la media de todos los objetos asignados al clúster. Esto significa que el centroide se desplaza hacia la media de los objetos del clúster. Este proceso se repite hasta que los centroides ya no se mueven de forma significativa o se cumple un criterio de convergencia.

Este criterio de convergencia cuyo cumplimiento implica la finalización de la repetición del proceso mencionado, se basa en que después de un número suficiente de iteraciones, los centroides convergerán a una posición estable y no se moverán de forma significativa en las siguientes iteraciones, es decir, el algoritmo habría encontrado una solución estable y los centroides ya no necesitarían ser recalculados.

El criterio de convergencia se define en función de una tolerancia de error, que indica el nivel de cambio permitido en la posición de los centroides. Si el cambio en la posición de los centroides es menor que la tolerancia de error, se considera que los centroides han convergido y el algoritmo se detiene.

En lugar de calcular la media de cada grupo como lo hace *K-Means*, *K-Modes* [58] utiliza la moda para determinar el centro de cada grupo, por este motivo, *K-Modes* ha sido elegido, pues el conjunto de datos a partir del cual se programa el algoritmo cuenta con variables de tipo categórico mayoritariamente.

El algoritmo *K-Modes*, a pesar de trabajar con datos de tipo categórico, requiere que estos sean numéricos, para ello previa programación del algoritmo o cualquier código relacionado con el mismo, ha sido necesario realizar transformaciones sobre los datos para codificarlos. [59] [60]

En este proyecto se han utilizado técnicas de codificación de variables categóricas, concretamente *Label Encoder* y *Ordinal Encoder*.

Label Encoder [61] se utiliza para transformar etiquetas categóricas en valores numéricos, el proceso de codificación consiste en asignar un número entero único a cada etiqueta categórica de la columna. Esta técnica se ha utilizado para la codificación de las variables: “Raza”, “Especie”, “Sexo”, “Tamaño” y “Microchip”.

Por otro lado, *Ordinal Encoder* [62] se utiliza para codificar variables de manera que los valores que se le asignen a la misma tengan en cuenta su orden natural. Una variable tiene orden natural cuando sus valores pueden ser ordenados de manera lógica, es decir, que los valores puedan ser comparados entre sí pudiendo determinar cuál es mayor o menor. Esto ocurre en el caso de las variables “Edad” y “Tiempo” del conjunto, y por eso se ha utilizado esta técnica para codificar estas variables.

Con el conjunto de datos codificado como correspondía en cada caso según lo explicado anteriormente, lo siguiente fue seleccionar el número óptimo de clusters. Para ello se utilizaron

dos métodos diferentes, en primer lugar, *el método de la silueta*, y en segundo lugar *el método del codo*. [63]

El método de la silueta [65] se basa en el cálculo de la silueta para cada punto de datos. La silueta es una medida de la similitud del punto con su propio clúster en comparación con los otros clústeres. La silueta toma valores entre -1 y 1, donde valores cercanos a 1 indican que el punto es muy similar a los puntos de su propio clúster, mientras que valores cercanos a -1 indican que el punto es muy diferente a los puntos de su propio clúster.

Para aplicar el *método de la silueta*, se calculó la silueta media para diferentes valores de k o diferentes números de clústeres (de dos a diez en este caso) y se visualizó una gráfica en función de k. Se escogió como óptimo el valor de k que maximiza la silueta media, lo cual indica que los clústeres están bien definidos y que los puntos dentro de cada clúster son similares entre sí.

Las puntuaciones medias obtenidas del *método de la silueta* no fueron tan buenas lo cual puede ser debido a que la efectividad de este método es mayor cuanto más irregulares sean las formas y los tamaños del clúster, tratándose de unos clusters esféricos y de tamaño más bien similares, se considera más eficiente la elección del número de clusters en base al *método del codo*, descrito a continuación. [64]

El *método del codo* [66] se basa en el cálculo de la suma de los errores cuadráticos dentro de los clústeres (SSE) en función del número de clústeres. Se grafica el SSE en función del número de clústeres, y se busca el punto de inflexión o "codo" en la curva, que indica el número óptimo de clústeres. El punto de inflexión es aquel donde añadir un clúster adicional no proporciona una mejora significativa en la calidad de la agrupación.

Los resultados del método del codo dieron que el número óptimo de clusters fue ocho, por lo que lo siguiente que se determinó fue el mejor tipo de inicialización entre "Cao", "Huang" y "Random" [67]. Los tipos de inicialización de *K-Modes*, son diferentes algoritmos que determinan cómo se seleccionan los centroides iniciales para los clusters.

Para seleccionar el mejor tipo de inicialización, se comparó el costo o suma de las distancias cuadradas de cada punto al centroide de su respectivo clúster de los diferentes tipos de inicialización, y el tipo de inicialización con menor coste fue el que se consideró como mejor para el conjunto de datos.

La tabla que se muestra a continuación resume las diferencias entre cada tipo de inicialización probado con sus respectivos costes, todos ellos probados para un número k de 8 clusters:

Tipo de inicialización	Funcionamiento	Coste
<i>Huang</i>	Selección de centroides en base a la distancia y la densidad. El primer centroide se selecciona de forma aleatoria y los siguientes se van seleccionando en	393791.0

	función de la distancia a los centroides existentes y la densidad local, pero priorizando siempre la distancia.	
<i>Cao</i>	Selección de centroides también teniendo en cuenta la densidad y la distancia, pero en este caso se prioriza la densidad ante la distancia.	418029.0
<i>Random</i>	Se seleccionan los centroides de forma completamente aleatoria, suele ser menos efectivo que los métodos de inicialización <i>Huang</i> y <i>Cao</i> .	402490.0

Tabla 6: *Tipos de inicialización en clustering*

Para las anteriores explicaciones del funcionamiento de cada tipo de inicialización, es importante definir los conceptos densidad y distancia para el contexto del clustering de datos: La distancia es una medida de la separación entre dos puntos en un espacio de características, y la densidad se refiere a la cantidad de puntos que hay alrededor de un centroide, si la densidad es alta significa que hay muchos puntos cercanos entre sí mientras que si la densidad es baja significará que los puntos están más dispersos.

Posteriormente, se entrenó y se guardó el modelo con el número *k* de clusters 8 y el tipo de inicialización *Huang*, que fue el mejor tal y como se concluyó en la tabla anterior.

3.3.3.2 Algoritmo de filtrado colaborativo basado en ítems para la recomendación de animales a usuarios

Otra de las funcionalidades más importantes del proyecto, es la implementación de un algoritmo de inteligencia artificial que recomiende animales compatibles con usuarios en base a la realización de un test compuesto por preguntas relacionadas con hábitos, estilos de vida y preferencias del mismo.

Esta solución, como se ha comentado en apartados previos de la memoria, agiliza las entrevistas a realizar por miembros de la protectora a candidatos adoptantes con la finalidad de comprobar si los entrevistados son compatibles con los animales o no. La implementación de esta funcionalidad en la aplicación permite que el usuario realice el cuestionario sin necesidad de ser supervisado por personal de los centros, y devuelve directamente el listado de animales compatibles entre los cuales el candidato adoptante podrá elegir su futura mascota.

Para el desarrollo de esta funcionalidad, se ha programado un algoritmo de recomendaciones de tipo filtrado colaborativo [68]. Este tipo de algoritmo utiliza la información de preferencia de los usuarios para sugerirles elementos que puedan ser de su interés, y por lo general puede tener dos grandes enfoques [69]:

- *Filtrado colaborativo basado en usuarios*: En este caso, el algoritmo utiliza información histórica de interacciones entre usuarios y elementos para generar recomendaciones. Más concretamente, analiza todos los posibles usuarios tratando de encontrar similitudes entre cada uno de los casos y el usuario objetivo, y en base a ellas recomendar elementos que hayan sido previamente recomendados con éxito a otros usuarios.

Este tipo de filtrado no es el que se corresponde con la funcionalidad a desarrollar debido a que además de que no se cuenta con un registro histórico de datos de usuarios, los datos que sean introducidos por los nuevos, no se almacenan en ningún lugar.

Por este motivo, se utilizará el segundo enfoque del algoritmo de recomendaciones de filtrado colaborativo.

- *Filtrado colaborativo basado en ítems*: En este otro caso, las recomendaciones se basan directamente en las similitudes entre las preferencias del usuario objetivo (los resultados del cuestionario en este caso), y los ítems a recomendar (los registros de animales).

Dado a que los datos de entrenamiento del algoritmo son únicamente los pertenecientes al conjunto de animales en estado de adopción, este enfoque es el que ha sido utilizado para la implementación de la funcionalidad de recomendaciones de animales para usuarios.

El algoritmo seleccionado, permite identificar las puntuaciones obtenidas del cuestionario rellenado por el usuario como un nuevo registro, y en base al mismo busca ítems o animales similares de manera que se obtenga un listado con los animales más compatibles con cada persona que realice el test [70].

Para la programación del algoritmo, de nuevo se ha utilizado *Python* y su librería *Scikit-learn*.

Para el procedimiento seguido, ha sido necesario en primer lugar el remuestreo de los datos. Dado que la cantidad de registros de la población generada era tan grande, el entrenamiento del algoritmo se interrumpía con excepciones de memoria puesto que los recursos no alcanzaban a soportar el volumen de datos de la población íntegramente.

Para solucionar este problema, se programó un intento de reducción de dimensionalidad por medio de la técnica PCA [72], también fallido puesto que ni siquiera se soportaba el volumen de datos tras esta reducción.

La técnica PCA [71] o de Análisis de Componentes Principales, se utiliza para reducir el número de variables en un conjunto de datos perdiendo la menor cantidad de información posible. Para esta reducción del número de variables, PCA transforma los datos iniciales en un conjunto de “componentes principales” no correlacionados, que a su vez son combinaciones lineales de las variables originales y representan la mayor cantidad posible de varianza de los datos [73]. Estos “componentes principales” se ordenan de mayor a menor varianza representada de los datos.

Como ya se ha mencionado, tras emplear la técnica PCA para reducir la dimensionalidad de los datos de la población de animales en adopción, la excepción de memoria seguía ocurriendo, por

lo que se optó por remuestrear el conjunto de datos para poder entrenar el modelo con una menor cantidad de registros.

Lo próximo a realizar fue la repartición de los datos en dos subconjuntos [74], siendo el 80% de los datos los que se dedicaron al entrenamiento de la muestra y el 20% restante a la evaluación del rendimiento del modelo. Los porcentajes de la división fueron escogidos en base a su popularidad en el área del aprendizaje automático, esta división proporciona una buena estimación del rendimiento del modelo en datos no vistos y ayuda a evitar el sobreajuste del modelo.

En pasos siguientes, se seleccionaron las variables a utilizar en el modelo, siendo estas únicamente las categorías cuyos valores fueran puntuaciones, ya que son las que verdaderamente interesan para el algoritmo como ya se ha explicado.

Posteriormente, se normalizaron los datos de las variables seleccionadas previamente con la finalidad de que todas las variables tuvieran una escala común. El modo en el que se realizó esta normalización fue transformando los datos de todas las variables de manera que estos se comprendieran en un rango de cero a uno, restando el valor mínimo de cada variable y dividiendo entre la diferencia del valor mínimo y el máximo [75].

Antes del entrenamiento del modelo y después de normalizar los datos, fue necesario obtener una medida de similitud entre los diferentes ítems que conforman el conjunto de datos. A pesar de que existen diferentes tipos de matrices de similitud que se pueden aplicar a los datos para algoritmos de filtrado colaborativo, para concretamente el enfoque basado en ítems es común la utilización de la similitud del coseno. Esta medida se basa en el ángulo de dos ítems, representados como vectores de características.

Por último, el modelo fue entrenado con los datos de entrenamiento, y fue testado con los datos de pruebas. A partir de las predicciones obtenidas, los resultados fueron evaluados utilizando dos métricas diferentes:

Nombre	Descripción	Resultado
Error cuadrático medio raíz (RMSE)	Mide la diferencia promedio entre los valores predichos y los valores reales dando mayor peso a los errores grandes que puedan impactar más sobre la calidad de las predicciones. Un error bajo indica que el modelo está realizando buenas predicciones.	0.02781155496033359
Error absoluto medio (MAE)	Mide de nuevo la diferencia promedio entre los valores predichos y los reales, pero es una medida lineal, trata a todos los errores por igual sin dar más peso a	0.01367477884430767

	los errores grandes. Un error bajo indica nuevamente que el modelo está realizando buenas predicciones.	
--	---	--

Tabla 7: *Tipos de métricas en filtrado colaborativo*

**Cabe destacar que, para un caso de uso de filtrado colaborativo con enfoque basado en usuarios, hubiera sido posible la elaboración de más métricas diferentes que pudieran aportar más información sobre los resultados del modelo.*

Ambos modelos fueron guardados para ser utilizados posteriormente dentro de funciones definidas en el *backend* de la aplicación que ajustasen las salidas de los algoritmos a los datos buscados en cada caso.

3.3.4 Desarrollo del Backend – FASE 4

Con la intención de comunicar las entradas recibidas por el usuario mediante la interfaz programada en el *frontend* de la aplicación con los modelos de inteligencia artificial entrenados y guardados en la fase anterior, se configuró un servidor [77] que además de recibir los datos de entrada y enviar los datos de salida, aplicó transformaciones a los datos por medio de funciones definidas explicadas en detalle posteriormente en este mismo apartado de la memoria.

Para esto se implementó un servidor web con *FastApi* [76] en *Python* en un script de extensión “py” donde se definió la lógica de la aplicación mediante funciones con las rutas necesarias especificadas en cada caso para recibir las peticiones procedentes de la aplicación de Android y procesarlas utilizando los algoritmos y otras funciones.

Por otro lado, en Android Studio, se programaron funciones para realizar peticiones HTTP al servidor, en estas peticiones se enviaron las variables introducidas por el usuario para cada funcionalidad de la aplicación, al servidor, y para ello se ha utilizado la biblioteca *Dio* [78] para el lenguaje de programación *Dart*.

Para la inicialización del servidor, se tuvo que establecer en el archivo un código indicando la IP y el puerto que fueran a ser utilizados. La IP seleccionada fue la dirección IP del sistema, que varía según la configuración de red; como esta IP también se tuvo que indicar dentro de las funciones de realización de peticiones de Android Studio, se creó una clase dentro de la aplicación para modificar la IP en todos los usos de la misma según el caso.

Por otro lado, para la elección del puerto teniendo en cuenta que este podría ser cualquiera entre los números del 0 al 65535 tal y como se aprendió en la asignatura de cuarto curso del grado “Virtualización y Seguridad”, simplemente se comprobó que estuviera disponible y no siendo utilizado por otro servicio en el sistema. También se evitaron puertos de privilegio, los cuales requieren permisos administrativos (de superusuario) para ser utilizados.

Para la ejecución del servidor se utilizó la consola de comandos de *Jupyter* (ya que es donde está instalado *Python* en el equipo de desarrollo), y cabe destacar la importancia de “matar” el servicio una vez finalizado para poder volver a utilizar el puerto.

Como ya se ha mencionado, además de la configuración del servidor en este mismo archivo se han programado funciones para transformar los datos de entrada en los de salida, también recibirlos y enviarlos. A continuación, se describen los objetivos de tales funciones.

- **Función de prueba:** Para comprobar la correcta ejecución del servidor, se definió una función de prueba sin ninguna utilización en el proyecto. Esta función simplemente enviaba un pequeño elemento de tipo JSON a la ruta definida de manera que, al acceder a la misma, en caso de que se hubiera ejecutado correctamente el servidor, aparecería la respuesta.

Una vez haber comprobado el correcto funcionamiento del servidor, se programaron el resto de las funciones:

- **Función para los datos del cuestionario de centros:** Para recoger, en primer lugar, los datos introducidos por el usuario una vez que este mismo pulsa el botón “enviar” de la aplicación en la pantalla correspondiente a la funcionalidad de recomendar centros. Los datos entran en la función en formato Json, y sobre estos valores se aplican diferentes transformaciones hasta alcanzar que estos datos de entrada sigan el mismo patrón que los datos del conjunto de animales adoptados correspondiente al registro histórico de adopciones de cada centro.
Se añade el registro del usuario como uno nuevo al conjunto mencionado, se aplica sobre este conjunto el algoritmo de agrupación, y la salida del mismo se basa en la adición de una nueva columna al conjunto de datos que mediante un atributo identificador determina a qué división pertenece cada registro.
En función del clúster al que pertenece el registro proporcionado por el usuario, se examina dentro de este mismo, la frecuencia de apariciones de cada identificador de centros para todos los registros de animales. Se fusionan los identificadores con el conjunto de datos de información de centros, y de este modo se obtienen los centros en los que el animal introducido por el usuario tendría más probabilidades de ser adoptado.
Esta función, se basa en la idea de buscar perfiles similares al del animal introducido por el usuario, y ver dónde han sido adoptados más animales como el objetivo.
Una vez conseguido el output deseado en formato *DataFrame*, se convierte en objeto de tipo *JsonResponse* antes de enviar los datos de nuevo a la aplicación de Android.
- **Función para los datos del cuestionario de compatibilidad animal-usuario:** Los datos del usuario llegan en formato de cadena de caracteres, pero otra función anidada examina cada una de las respuestas en formato de texto proporcionadas por el usuario y le atribuye una puntuación a cada variable numérica de las presentes en el conjunto de datos de animales en adopción (categorías obtenidas del procesamiento de las descripciones de los animales).

Una vez la entrada de datos con variables de tipo cadena de caracteres es transformada en los datos decimales buscados para la entrada del modelo, se vuelven a aplicar un par de transformaciones sobre los mismos para normalizar los datos antes de pasarlos como entrada al algoritmo de recomendaciones.

Una vez se aplica el algoritmo, la salida que se obtiene del mismo además de la distancia (explicada en apartados anteriores de la memoria), es el índice de los elementos más cercanos en función de esta distancia calculada en función de las similitudes entre ítems. Por tanto, se seleccionan los índices obtenidos de la salida del algoritmo para el conjunto de datos de animales en adopción, y esos registros serán entonces los que coincidan en mayor medida con las respuestas proporcionadas por el usuario en el test.

De nuevo este output se presenta en formato *DataFrame* y para devolverlo a la aplicación al igual que en el anterior caso, se vuelve a convertir en un objeto de formato *JsonResponse*.

- Función para mostrar datos de centros: Programada para que cuando el usuario pulse el botón de la funcionalidad correspondiente en el inicio de la aplicación, se envíen todos los datos del conjunto de centros al frontend de modo que el usuario pueda visualizarlos correctamente.

Para ello se carga directamente el conjunto de datos de centros dentro de la función (en estructura *DataFrame*), y se convierte de nuevo en objeto de tipo *JsonResponse* ya que es el aceptado por *Flutter*.

Cada una de estas funciones, fueron programadas para ser ejecutadas cada vez que se realizase una solicitud HTTP de tipo GET a las rutas definidas en cada caso.

3.3.5 Desarrollo del Frontend – FASE 5

Tal y como ya se ha mencionado en otros apartados anteriores de la memoria, el entorno elegido para la programación del *frontend* de la aplicación móvil ha sido *Android Studio*, su correspondiente framework *Flutter* y su lenguaje de programación propio *Dart*.

En primer lugar, se realizó la instalación de todo lo necesario para utilizar el entorno, su framework y su lenguaje de programación, además de proporcionar los permisos necesarios para cada caso. También hubo que seleccionar un SDK, cuya elección ha sido “Android 11” en base a que este permite que la aplicación sea compatible con la gran mayoría de los dispositivos Android, incluso con los más recientes.

Para simular la ejecución del *frontend* en un dispositivo Android, se ha configurado un Emulador [79], es decir, una instancia del sistema operativo Android que permite probar la aplicación a medida que se va creando o modificando. El Emulador seleccionado ha sido un smartphone Pixel 2.

Para terminar de configurar el Emulador, se ha requerido especificar una “imagen del sistema”, que es un conjunto de archivos que representan una versión específica del sistema operativo de

Android en este caso. En base a las recomendaciones realizadas por el propio entorno de *Android Studio*, se ha seleccionado “system image R” como imagen del sistema.

En cuanto al diseño del *backend*, este se bocetó en primer combinando las herramientas Figma y Canva (enlace del boceto adjunto en el anexo), luego se trató de replicar en la aplicación de Android Studio cada uno de los elementos previamente diseñados, adaptándolos a las posibilidades del tiempo del que se dispone para realizar el proyecto actual.

Además de los elementos estáticos y botones diseñados en el boceto, se incluyeron en el *frontend* otros elementos como pantallas de espera para evitar excepciones visibles de cara al usuario hasta que no se terminasen de realizar las peticiones con el *backend*.

El apoyo fundamental para la programación del *frontend*, ha sido la documentación oficial de la herramienta utilizada, no obstante, para elementos concretos existen numerosos blogs, videotutoriales y repositorios de desarrolladores que enseñan como desarrollar fragmentos de código en concreto.

En cuanto a otras funcionalidades que no requiriesen del servidor desarrollado en *Python* con *FastApi*, se ha implementado un mecanismo de creación de cuentas e inicio de sesión para el usuario a través de *Firebase Authentication*.

Firebase Authentication [80] es uno de los servicios que conforman *Firebase*, la plataforma de Google para el desarrollo de aplicaciones en la nube. Este servicio en concreto facilita la autenticación de usuarios, permitiendo que esta se pueda realizar vía correo electrónico o usuario o contraseña, tal y como se ha implementado en este caso, aunque también permite la autenticación mediante proveedores externos (cuenta de Google, Facebook o Twitter entre otros).

Mediante la interfaz del proyecto de *Firebase*, se puede controlar el acceso de los recursos de la aplicación (al desarrollador), permitiendo que se puedan vetar, permitir o limitar el acceso según el usuario que se loguee.

La inclusión de *Firebase Authentication* en el proyecto ha sido relativamente sencilla dado que existen dependencias concretas en Android Studio justamente para implementar este tipo de funcionalidad. Además, se ha consultado un codelab¹⁴ para la integración de un proyecto de Android en la plataforma *Firebase*, en el que se explican perfectamente las instrucciones de instalaciones y ejecuciones pertinentes [81].

Para entender la conexión entre los resultados de las diferentes fases de desarrollo, se ha elaborado el siguiente diagrama que representa la arquitectura del proyecto:

¹⁴ “Un codelab es un instructivo breve y autónomo que guía al usuario para realizar una tarea específica.” (Becker, 2019) [82]

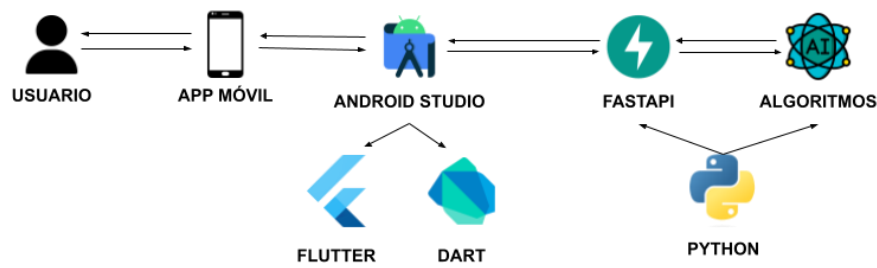


Figura 7: Diagrama de arquitectura del proyecto

Y con la finalidad de entender la utilidad del proyecto de manera clara, el siguiente diagrama de casos de uso:

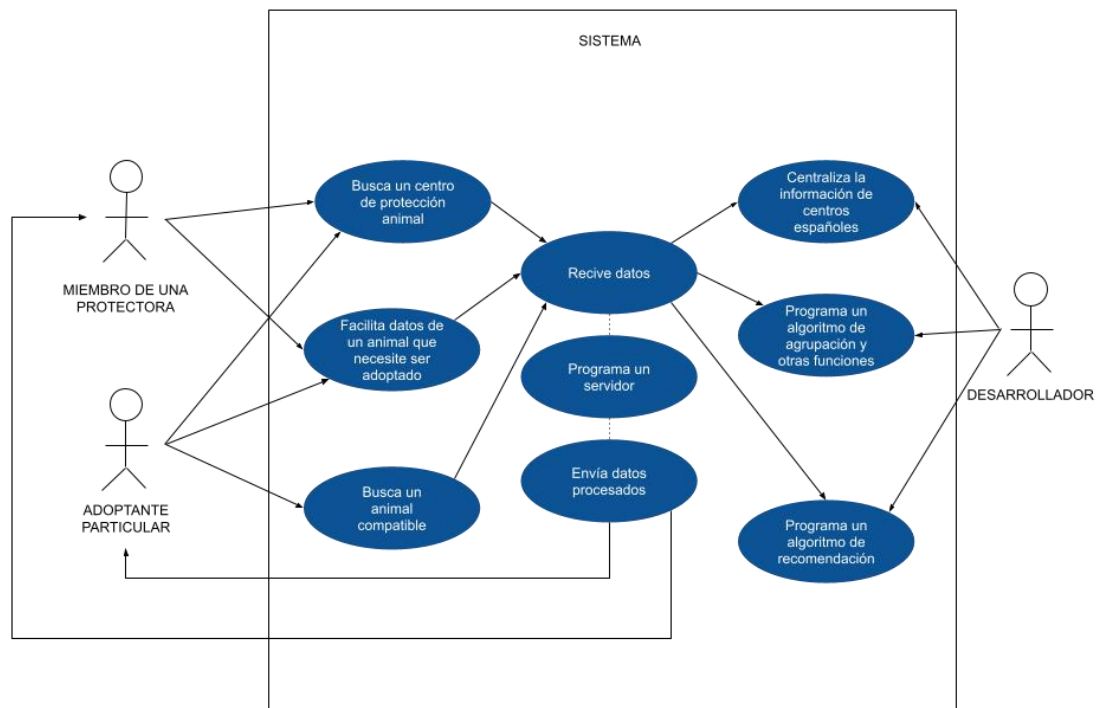


Figura 8: Diagrama de casos de uso

Capítulo 4. DISCUSIÓN

Los objetivos establecidos inicialmente en la propuesta del proyecto, fueron menores que los alcanzados finalmente debido a la falta de conocimiento y formación en las herramientas empleadas en un principio, pero a medida que se fueron superando las etapas de desarrollo, el conocimiento y la involucración fue aumentando a la vez que lo fue haciendo la magnitud de los objetivos.

Lo primero que se buscó en el desarrollo del proyecto, fue asegurar el cumplimiento de los objetivos establecidos inicialmente, pero, a medida que las tareas se iban desempeñando exitosamente y por la motivación del proyecto, se fueron investigando nuevas alternativas, herramientas y soluciones para añadir a cada funcionalidad de la aplicación desarrollada y así sucesivamente. Por tanto, se puede decir que la diferencia entre la complejidad del contenido de las tareas planificadas y las desarrolladas ha sido abismal por los motivos mencionados.

Del modo que el hecho de ir desarrollando tareas ha provocado la intención de querer mejorarlas o perfeccionarlas constantemente, ha llegado un momento en el que se han tenido que ir acotando y concretando las limitaciones del proyecto, ya que el tiempo del que se dispone para su elaboración de lo contrario podría resultar una amenaza.

La aplicación nació de un caso de uso real, en el que una conocida me comentaba que había tenido que acudir a un centro de protección animal de una aldea periférica ya que los animales de las protectoras del centro de la ciudad eran demasiado grandes. En ese momento se planteó la siguiente pregunta ¿Cuántas personas habrán dejado pasar la adopción de un animal porque en su centro de protección animal más cercano no existiera un ejemplar con las características deseadas? A lo que la respuesta fue Adoption Movement, si los adoptantes no se trasladan con tanta facilidad a otros lugares geográficos para adoptar, quizá la clave sean los traslados de animales sabiendo que haya lugares donde será más probables que sean adoptados que en otros según sus características.

A partir de la idea descrita en el párrafo anterior, surgieron más casos de uso en los que la funcionalidad de la aplicación podría ser útil del mismo modo, como por ejemplo, centros de protección saturados con la necesidad de despejar sus instalaciones garantizando que no van a contribuir así a la saturación a su vez, de los centros a los que se trasladen los animales desahuciados, o bien particulares que no se pudieran hacer cargo de sus mascotas por problemas personales, incompatibilidades o camadas indeseadas entre otros motivos.

El hecho de haber entrado en contacto recientemente con el mundo del desarrollo de aplicaciones móviles motivó la idea de desarrollar el sistema mencionado de este modo, lo cual fue completamente acertado debido a que es una de las maneras más adecuadas de apoyar a la necesidad que cubre el proyecto.

Lo conseguido finalmente con este proyecto, ha sido un prototipo de aplicación con datos mayoritariamente ficticios o sintéticos, pero que demuestra la utilidad de la solución si se aplicara en el futuro a datos reales.

Capítulo 5. CONCLUSIONES

5.1 Conclusiones del trabajo

Tanto el proyecto en sí como el estado del arte y antecedentes del mismo han confirmado las carencias que tiene el ámbito de la protección animal en España.

Por un lado, la Inteligencia Artificial, cada vez más presente en todas y cada una de las áreas entre las que nos movemos en nuestro día a día, está demostrando cada vez más la capacidad que presenta de poner soluciones a problemas de una manera impresionantemente eficiente.

Lo preocupante, es que estos avances no hayan llegado, o al menos de momento, a ayudar sin ánimo de lucro a entidades o asociaciones que velan por el bienestar ajeno sin esperar nada a cambio. Si la aplicación en un futuro pudiese a llegar a ser publicada para cualquier usuario, sería completamente gratuita a pesar de los costes que supondría al desarrollador reflejados en el presupuesto, eliminando cualquier pensamiento acerca de la monetización de la solución dado a que las personas que la utilizarían le sacarían partido remediando necesidades, y no aportando valor a necesidades ya cubiertas.

Por otro lado, resulta sorprendente la escasa digitalización presente en las herramientas de las asociaciones contactadas. En este proyecto, se ha contribuido a ella centralizando, además, los datos de los centros de protección animal españoles en un mismo lugar y manteniéndolos accesibles para cualquier público que se quiera registrar como usuario en Adoption Movement.

Uno de los inconvenientes principales del proyecto, ha sido la dificultad de acceder a datos reales por el motivo expuesto anteriormente, y por ello una de las posibles líneas futuras a seguir en el proyecto, se trata en la ampliación de las funcionalidades de la misma incluyendo la posibilidad de que los miembros de los centros puedan elaborar una base de datos de sus animales a través de una interfaz sencilla, y que pueda ser consultada cada vez que se necesite.

5.2 Conclusiones personales

La autoexigencia provocada por la realización del proyecto ha convertido al mismo en una tarea dura de realizar, pero, aun así, todas las conclusiones obtenidas en el ámbito personal son mayoritariamente positivas.

En primer lugar, el hecho de comprobar que lo aprendido durante el grado universitario cursado una vez puestos en práctica los conocimientos, despierta el sentimiento de entusiasmo y motivación que se persigue encontrar en tareas que se van a realizar rutinariamente, confirma que la elección de la titulación (de la que no estaba segura al cien por cien), ha sido completamente un acierto.

En segundo lugar, me enorgullece plenamente pensar en los conocimientos que tenía cuatro años atrás en la rama de la Informática, las Matemáticas y el Análisis de Datos en comparación con los adquiridos hasta día de hoy.

Por último, incluso habiendo adquirido durante los años de curso todos los conocimientos mencionados en el anterior párrafo, la visión personal de lo que podría ser capaz de hacer antes y después del desarrollo del proyecto ha sufrido extremas modificaciones en el proceso.

Por todo esto, y a pesar de que seguramente no vaya a ser el mejor proyecto que vaya a hacer, me siento plenamente satisfecha sobre todo con la inclinación de la curva de aprendizaje durante estos meses.

Capítulo 6. FUTURAS LÍNEAS DE TRABAJO

Como la principal limitación del proyecto actual ha sido el tiempo del que se disponía para su desarrollo, se ha tomado conciencia durante la elaboración del mismo de la posibilidad de realizar diversas mejoras o ampliar el alcance en caso de realizar una continuación e invertir más tiempo en ello.

Tal y como se ha explicado en la discusión y conclusiones del proyecto, a medida que se iban completando objetivos y tareas, iban surgiendo nuevas ideas de mejoras de las mismas o de la aplicación en general, con estas ideas se identifican las futuras líneas de trabajo del proyecto, enumeradas a continuación:

- Creación de otra **aplicación asociada**, o bien de una nueva funcionalidad de la misma (en ese caso habría que diferenciar tipos de usuarios entre particulares y miembros de protectoras), que permita por cada centro de protección animal, crear una base de datos que pueda ser modificada por usuarios de la aplicación sin demasiados conocimientos informáticos mediante una interfaz sencilla e intuitiva.
- En la base de datos mencionada, podría incluirse la posibilidad de que el usuario de la protectora pudiese fotografiar cada animal correspondiente al registro introducido (o bien cargar una foto del mismo), y por tanto se pudiera adaptar tanto el *backend* como el *frontend* de la aplicación para integrar en ellos el **manejo de imágenes introducidas por el usuario**.
- La anterior aplicación o funcionalidad asociada, permitiría a su vez la **obtención de datos reales de animales**, tanto históricos como en tiempo real, por lo que se cubriría a la vez otra de las necesidades y la mayor de las limitaciones del proyecto actual.
- Tal y como se ha explicado en anteriores apartados de la memoria, los datos de centros han sido reales al ser obtenidos mediante el listado de asociaciones de cada Comunidad Autónoma española. Aún así, algunos de los atributos del conjunto de datos final fueron generados con *fake data*, por lo que otra línea futura podría ser obtener todos los **atributos reales de cada centro** por medio de encuestas o bien dedicando más tiempo a la obtención de datos a mano.
- Las líneas futuras explicadas en puntos anteriores, sustituirían la tarea de la generación de poblaciones sintéticas para el entrenamiento de los modelos de inteligencia artificial, pero en caso de que no se pudieran lograr los puntos anteriores, otra futura línea de trabajo sería el **aumento de precisión de las distribuciones** empleadas para la generación de datos en base a las mismas, obteniendo más información de diferentes fuentes y no solamente de los informes de la *Fundación Affinity*.

- Una línea futura que sustituiría la subida o carga de imágenes por parte del usuario en caso de seguir sin poder obtener datos reales, sería la **generación de imágenes** con herramientas de Inteligencia Artificial para los logos de los centros por un lado, y la selección de imágenes acorde a las características del animal introducido a partir de un conjunto de datos de imágenes animales (por ejemplo, alguno de *Kaggle*¹⁵ propuesto para algún reto de clasificación de imágenes entre especies felinas y caninas)
- Otra posible línea de futuro del proyecto podría ser la **mejora del algoritmo de filtrado colaborativo basado en ítems**, combinando el enfoque actual con un filtrado colaborativo basado en usuario, para lo cual se necesitaría almacenar las respuestas de los usuarios para utilizarlas en otros resultados indirectamente.
- Si el anterior punto se llegara a alcanzar, tendrían que comenzar a **estudiarse temas legales de protección de datos** y probablemente hacer que el usuario aceptase permisos para poder seguir adelante con las funcionalidades de la aplicación, ya que del mismo modo que datos de otros usuarios se utilizarían para el entrenamiento del modelo, los datos del usuario objetivo también se emplearían para otras predicciones.
- En cuanto a la programación del **backend**, para el actual proyecto se han estudiado los conceptos más básicos del tema, pero en un futuro se podría profundizar más en su funcionamiento para **perfeccionarlo**.
- Desarrollo de una versión de la aplicación móvil para sistema **operativo iOS**, además del desarrollo actual para sistema operativo Android.

¹⁵ “Más de 50 000 conjuntos de datos públicos y 400 000 cuadernos públicos.”(Kaggle) [83]

Capítulo 7. GUÍA DE LA APLICACIÓN


La aplicación móvil de Adoption Movement, es compatible con dispositivos Android. No está desarrollada para iPhone en esta primera versión.

El uso de la aplicación se describe a continuación:

1. Pantalla de bienvenida: Permite al usuario navegar hasta la página de registro o la de inicio de sesión.



2. Registro: Se solicita al usuario el nombre, correo electrónico y contraseña para la creación de un perfil en *Firebase Authentication*, además desde esta pantalla se puede navegar directamente a la de inicio de sesión si el usuario ya tiene una cuenta.



3. Inicio de sesión: Donde el usuario debe introducir sus credenciales creadas en la pantalla de registro para poder acceder al contenido de la aplicación.



4. Pantalla de bienvenida: Se muestran tres botones correspondientes a cada una de las principales funcionalidades de la aplicación, también se permite cerrar sesión desde esta misma pantalla.



5. Primer botón “¿Necesitas que un animal sea adoptado?”: El botón dirige al usuario a una página donde deberá introducir los datos correspondientes al animal que desea dar en adopción, al pulsar el botón de enviar en la pantalla “Características del animal” si todos los campos están marcados, se dirige al usuario a una página con los resultados del algoritmo programado en el *backend*, “Centros recomendados”, que consta de un listado de x centros para cada caso, donde el animal con las características introducidas, tendrá mayor probabilidad de ser adoptado.

← Características del animal	← Centros recomendados
Especie Canina	1. RESCATE ANIMAL MARINA ALTA (R.A.M.A) Valencia
Raza Mestizo	2. Associació Progat Terrassa Cataluña
Tamaño Grande	3. ANIMALS SENSE SOSTRE D'ELX Valencia
Sexo <input checked="" type="radio"/> Macho <input type="radio"/> Hembra	4. ASOCIACION CRUZ AZUL PARA MASCOTAS Murcia
Edad Años Meses	5. Asociación Protectora de Animales Míamur nan Euskadi
Tiempo en adopción Años Meses	
Microchip <input type="checkbox"/>	
Enviar	

6. Segundo botón “Haz el test de compatibilidad”: Dirige al usuario a un conjunto de pantallas en las que se muestran diferentes preguntas, diez en total, que conformarán el input del algoritmo de recomendación programado en el *backend*.

Test de compatibilidad

1. ¿Incluirá a su mascota en actividades que supongan salir de casa periódicamente? (playa, vacaciones, actividades de fin de semana...)

☐ SI ☐ NO
 2. ¿Su animal de compañía se relacionará generalmente con su entorno y círculo de amistades o pasará más tiempo en su hogar?

☐ ENTORNO ☐ HOGAR
 3. ¿Cuánto tiempo pasará en casa o junto con el animal, la mayor parte del día o periodos cortos en horarios intermitentes?

☐ LA MAYOR PARTE DEL DÍA ☐ PERIODOS CORTOS
 4. ¿Convive con niños o tiene pensado hacerlo en un futuro?

☐ SI ☐ NO
 5. ¿Su residencia cuenta con alguna zona en la que su mascota se pueda mover libremente? (jardín, patio, finca...)

☐ SI ☐ NO

Siguiente

Test de compatibilidad

6. ¿Tiene o estaría dispuesto a tener licencia PPP?

☐ SI ☐ NO
 7. ¿Preferiría un animal cariñoso y que busque afecto constantemente o uno que sea más independiente?

☐ CARIÑOSO ☐ INDEPENDIENTE
 8. ¿Tiene actualmente o tendría pensado tener en un futuro otros animales?

☐ SI ☐ NO
 En caso afirmativo, ¿puede especificar más?

Siguiente

Test de compatibilidad

9. Haga un ranking en función de los aspectos positivos que más valore en un animal. Pulse las flechas para desplazar los ítems.

1 Lealtad
2 Alegría
3 Inteligencia
4 Fortaleza
5 Belleza
6 Capacidad de adaptación

Siguiente

Test de compatibilidad

10. Haga un ranking en función de los aspectos negativos que más rechaza le generen hacia un animal. Pulse las flechas para desplazar los ítems.

1 Brutalidad
2 Tristeza
3 En recuperación (enfermedad)
4 Historial de enfermedades
5 Víctima de maltrato
6 Suelta pelo

Siguiente


7. **Animales recomendados:** Los resultados del test de compatibilidad se muestran en forma de diferentes páginas que conforman un listado de los cien animales que mejor se adaptarían al usuario en función de los datos proporcionados en el cuestionario previo. Para cada registro se puede obtener más información sobre el mismo pulsando encima de cada elemento del listado.

Animales recomendados

1. Juan David
Cruce Mastín
 2. Miguel
Yorkshire Terrier
 3. Juan Felipe
Cruce Pastor del Cáucaso
 4. Juan Felipe
Podenco
 5. Juan Diego
Cruce Podenco
 6. Enrique
Cruce Podenco

← 1/17 →

Detalles del Animal



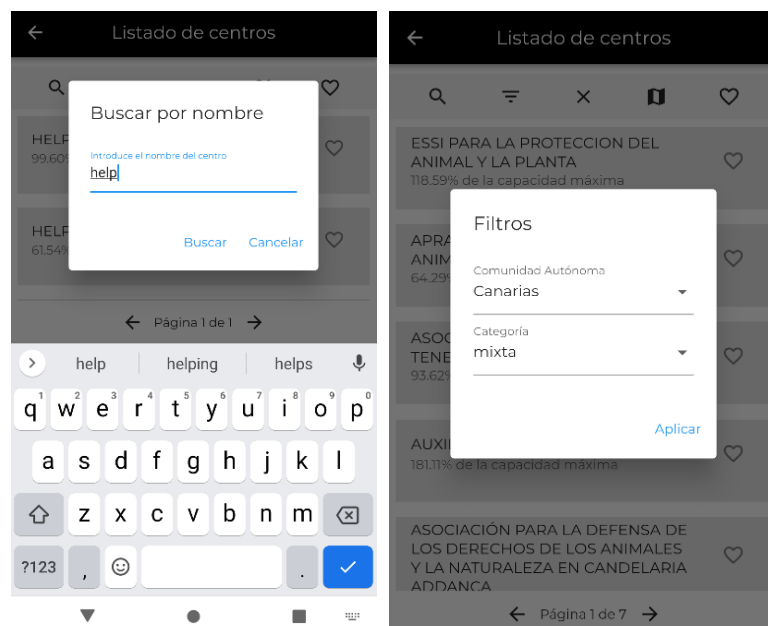
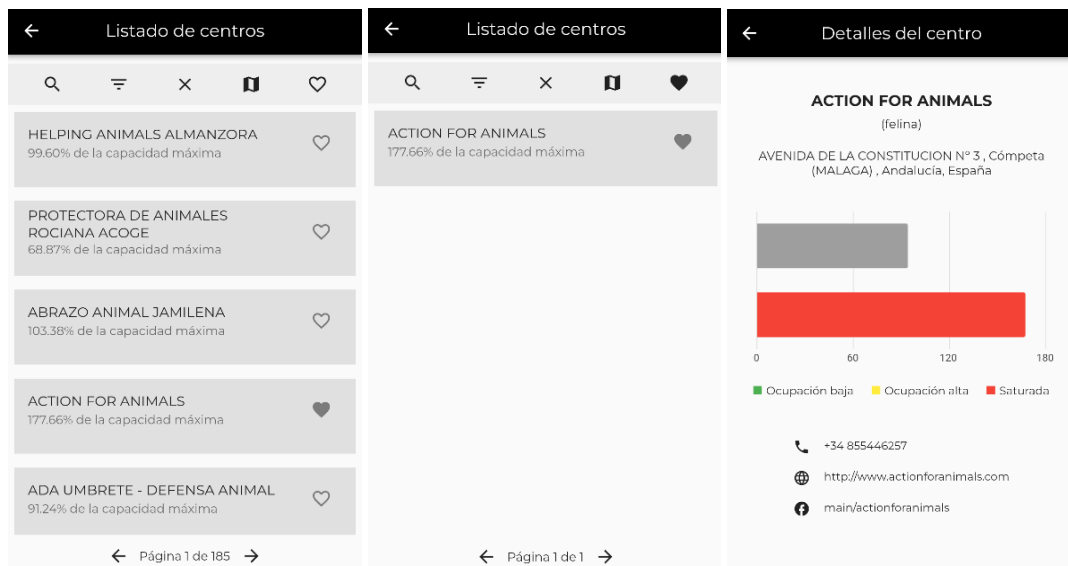
Juan Felipe
Cruce Pastor del Cáucaso

♂ Macho

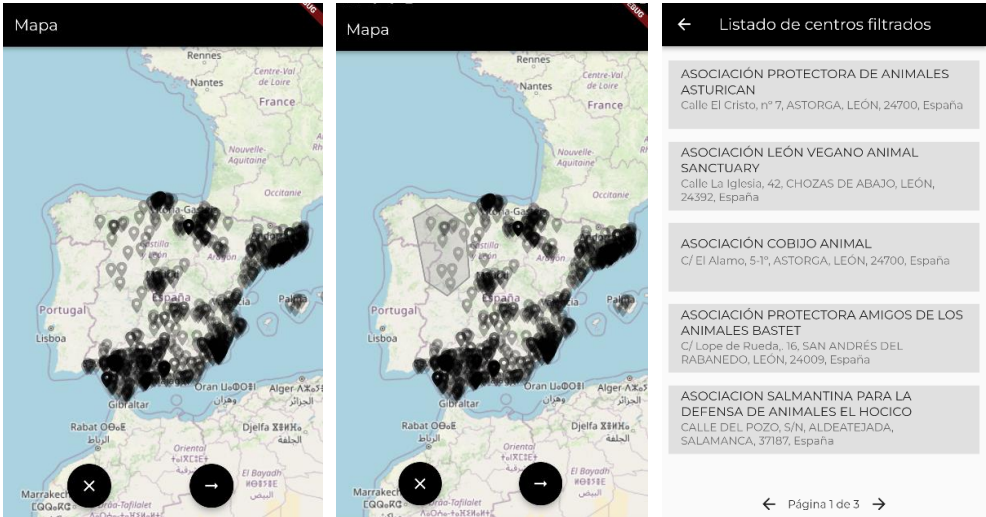
🎂 3 años

Está en...
Associació Protectora dels Animals del Moianès (Cataluña)

8. Tercer botón “Listado de centros”: Dirige al usuario a un listado de todos los centros de protección animal que constan en la base de datos de la aplicación permitiendo aplicar diferentes filtros sobre los mismos, así como marcar centros como favoritos o bien obtener más información de cada protectora pulsando sobre cada elemento de la lista.



9. Mapa: Además de los filtros ilustrados en el anterior punto, existe un tercer filtro que dirige a una pantalla con un mapa, donde el usuario puede dibujar el área en el que desea enfocar su búsqueda.



- [14] (RedHat, 2020). “¿Qué es un SDK?” Obtenido de:
<https://www.redhat.com/es/topics/cloud-native-apps/what-is-SDK>
- [15] (carlosbermudez, 2011). “Raw Data: definición de hoy” Obtenido de:
<https://www.digitalika.com/2011/08/raw-data-definicion-de-hoy/>
- [16] (Convierte PDF a Excel) https://www.ilovepdf.com/es/pdf_a_excel
- [17] (Ciberseguridad, s.f.). “¿Qué es el Web Scraping y para qué se utiliza?” Obtenido de:
https://ciberseguridad.com/guias/recursos/web-scraping/#%C2%BFQue_es_el_web_scraping
- [18] Muthukadan, B. “Selenium with Python” Obtenido de:
<https://seleniumpython.readthedocs.io/>
- [19] (Selenium) “Selenium 4.8 documentation” Obtenido de:
<https://www.selenium.dev/selenium/docs/api/py/api.html>
- [20] (Google Sites) “ChromeDriver- WebDriver for Chrome” Obtenido de:
<https://sites.google.com/a/chromium.org/chromedriver/downloads>
- [21] (Python) “Python 3.11.3 documentation” Obtenido de: <https://docs.python.org/3/>
- [22] (Amazon Web Services, s.f.) “¿Qué es ETL?” Obtenido de:
[https://aws.amazon.com/es/whatis/etl/#:~:text=Extracci%C3%B3n%2C%20transformaci%C3%B3n%20y%20carga%20\(ETL\)%20es%20el%20proceso%20consistente,central%20llamado%20al%20macenamiento%20de%20datos.](https://aws.amazon.com/es/whatis/etl/#:~:text=Extracci%C3%B3n%2C%20transformaci%C3%B3n%20y%20carga%20(ETL)%20es%20el%20proceso%20consistente,central%20llamado%20al%20macenamiento%20de%20datos.)
- [23] (Arimetrics, s.f.) “Qué es Script” Obtenido de:
<https://www.arimetrics.com/glosariodigital/script#:~:text=Qu%C3%A9%20es%20Script,concreto%20para%20herramientas%20en%20internet>
- [24] (Aprende con Alf, s.f.) “La librería Pandas” Obtenido de:
<https://aprendeconalf.es/docencia/python/manual/pandas/#:~:text=Un%20objeto%20del%20tipo%20DataFrame,contender%20datos%20de%20distintos%20tipos>
- [25] (Pandas) “Pandas documentation” Obtenido de: <https://pandas.pydata.org/docs/>
- [26] (OpenStreetMap, s.f.) “OpenStreetMap” Obtenido de:
<https://www.openstreetmap.org/about>
- [27] Cera, R. (2019). “El fake data es mucho más peligroso que las fake news” Obtenido de:
<https://www.puromarketing.com/12/31973/fakedatamuchomaspeligrosorefakesnews#:~:text=Un%20fake%20data%20es%20aquel,es%20consecuencia%20de%20donde%20procede>

- [28] "Welcome to GeoPy's documentation!" Obtenido de:
<https://geopy.readthedocs.io/en/stable/>
- [29] Morales, A. (s.f.) "Cómo realizar geocodificación con GeoPy" Obtenido de:
<https://mappinggis.com/2018/11/geocodificacion-con-geopy/>
- [30] "Welcome to Faker's documentation!" Obtenido de:
<https://faker.readthedocs.io/en/master/>
- [31] "random- Generar números pseudoaleatorios" Obtenido de:
<https://docs.python.org/es/3/library/random.html>
- [32] "re- Regular expression operations" Obtenido de:
<https://docs.python.org/3/library/re.html>
- [33] Gómez-Calcerrada, S. (2015) "Guía para entender y usar expresiones regulares" Obtenido de: <https://www.adictosaltrabajo.com/2015/01/29/regexsam/>
- [34] (Editorial, 2022) "Guau! Qué perros, la nueva app que fomenta la adopción" Obtenido de:
<https://misanimales.com/nueva-app-fomenta-adopcion/>
- [35] Bukosabino (2016) "Guau! qué perros" Obtenido de :
<https://apkpure.com/guauqu%C3%A9perros/com.ionicframework.buscaperrocliente620403>
- [36] Post de la red social *Instagram* publicado por su propio creador, la aplicación *AdoptaMe*. Obtenido de: https://www.instagram.com/p/COnbwFlnTbZ/?utm_source=ig_embed
- [37] Coca, L. (2016) "AdoptaMe, la aplicación que permite adoptar animales en muy pocos pasos" Obtenido de: https://los40.com/los40/2021/05/15/bigbang/1621072918_248148.html
- [38] (Última hora, 2016) "Nace Miwuki, una aplicación que facilita adoptar animales de protectoras de todo el mundo" Obtenido de:
<https://www.ultimahora.es/noticias/sociedad/2017/02/15/248810/nacemiwukiaplicacionfacilita-adoptar-animales-protectoras-todo-mundo.html>
- [39] (Miwuki) Cuestionario para los "matches" en adopciones. Obtenido de:
<https://petshelter.miwuki.com/matches>
- [40] (Tinder) "Tinder" Obtenido de: <https://tinder.com/es-ES>
- [41] Tupper, S. (2019) "'GetPet': La nueva app para encontrar perros en adopción" Obtenido de:
<https://www.rockandpop.cl/2019/02/getpet-la-nueva-app-para-encontrar-perros-en-adopcion/>
- [42] Shah, P. (2020) "Sentiment Analysis using TextBlob" Obtenido de:
<https://towardsdatascience.com/my-absolute-go-to-for-sentiment-analysis-textblob->

[3ac3a11d524#:~:text=TextBlob%20returns%20polarity%20and%20subjectivity,help%20with%20fine%2Dgrained%20analysis.](#)

[43] La La, F. (2018) “Inteligencia artificial: Análisis de sentimiento de textos” Obtenido de: <https://learn.microsoft.com/es-es/archive/msdn-magazine/2018/may/artificially-intelligent-text-sentiment-analysis>

[44] (Wikipedia, 2019) “Palabra vacía” Obtenido de: https://es.wikipedia.org/wiki/Palabra_vac%C3%ADa

[45] (NLTK) “NLTK documentation” Obtenido de: <https://www.nltk.org/>

[46] (Scikit learn) “sklearn.feature_extraction.text.TfidfVectorizer” Obtenido de: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[47] (Sefidian Academy) “Understanding TF-IDF with Python example” Obtenido de: <http://www.sefidian.com/2022/07/28/understanding-tf-idf-with-python-example/>

[48] (Unioviedo, s.f.) “El algoritmo k-means aplicado a clasificación y procesamiento de imágenes” Obtenido de: https://www.unioviedo.es/compnum/laboratorios_py/kmeans/kmeans.html#kmeans

[49] (Gnesim, s.f.) “Word2Vec embeddings” Obtenido de: <https://radimrehurek.com/gensim/models/word2vec.html>

[50] (Scikit learn) “sklearn.cluster.KMeans” Obtenido de: <https://scikitlearn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

[51] Moya, R. (2016) “Selección del número óptimo de Clusters” Obtenido de: <https://jarroba.com/seleccion-del-numero-optimo-clusters/>

[52] (Fundación Affinity) “Infografía Él nunca lo haría. Estudio de abandono y adopción 2022” Obtenido de: <https://www.fundacion-affinity.org/observatorio/infografia-el-nunca-lo-haria-abandono-adopcion-perros-gatos-espana-2022>

[53] (NumPy) “NumPy Documentation” Obtenido de: <https://numpy.org/doc/>

[54] (Jiménez, 2019) “El módulo array frente a las listas Python” Obtenido de: <https://python-paraimpacientes.blogspot.com/2019/07/elmoduloarrayfrentelaslistas.html#:~:text=El%20m%C3%B3dulo%20array%20de%20la,de%20punto%20flotante%2C%20entre%20otros.>

[55] (Mimesis) “Mimesis: Fake Data Generator” Obtenido de: <https://mimesis.name/en/master/>

[56] (Wikipedia, 2023) “Modelo entidad-relación” Obtenido de: https://es.wikipedia.org/wiki/Modelo_entidad-relaci%C3%B3n

[57] (Python package documentation, 2022) “Python implementations of the k-modes and k-prototypes clustering algorithms for clustering categorical data.” Obtenido de: <https://pypi.org/project/kmodes/>

- [58] Ramirez, J. (2020) "K-Modes Algorithm" Obtenido de:
<https://medium.com/@jonathanrmzg/k-modes-algorithm-d4b54aca1e01>
- [59] Sarkar, D. (2018) "Categorical Data" Obtenido de:
<https://towardsdatascience.com/understanding-feature-engineering-part-2-categorical-data-f54324193e63>
- [60] Brownlee, J. (2020) "3 Ways to Encode Categorical Variables for Deep Learning" Obtenido de: <https://machinelearningmastery.com/how-to-prepare-categorical-data-for-deep-learning-in-python/>
- [61] (Scikit learn) "sklearn.preprocessing.LabelEncoder" Obtenido de:
<https://scikitlearn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>
- [62] (Scikit learn) "sklearn.preprocessing.OrdinalEncoder" Obtenido de:
<https://scikitlearn.org/stable/modules/generated/sklearn.preprocessing.OrdinalEncoder.html#sklearn.preprocessing.OrdinalEncoder>
- [63] (Data Novia, s.f.) "Determining The Optimal Number Of Clusters: 3 Must Know Methods" Obtenido de: <https://www.datanovia.com/en/lessons/determining-the-optimal-number-of-clusters-3-must-know-methods/>
- [64] (Python Data Science Handbook, s.f.) "In Depth: k-Means Clustering" Obtenido de:
<https://jakevdp.github.io/PythonDataScienceHandbook/05.11-k-means.html#Evaluating-KMeans-clustering>
- [65] (Scikit learn) "Silhouette Coefficient" Obtenido de:
<https://scikitlearn.org/stable/modules/clustering.html#silhouette-coefficient>
- [66] (Wikipedia, 2023) "Elbow method (clustering)" Obtenido de:
[https://en.wikipedia.org/wiki/Elbow_method_\(clustering\)](https://en.wikipedia.org/wiki/Elbow_method_(clustering))
- [67] Jiang, F., Liu, G., Deu, J. y Sui, Y. (2016) "Initialization of k-modes clustering using outlier detection techniques" Obtenido de:
<https://ccc.inaoep.mx/~ariel/2016/2016%20Initialization%20of%20kmodes%20clustering%20using%20outlier%20detection.pdf>
- [68] Saavedra, J. (2017) "Sistemas de recomendación, parte 2: filtrado colaborativo" Obtenido de: <https://medium.com/@eng.saavedra/sistemas-de-recomendaci%C3%B3n-parte-2-b8a5dc9dc730>
- [69] Pérez, A. (2017) "Sistemas de recomendación" Obtenido de:
<https://recommendersys.wordpress.com/2017/10/20/tipos-de-rs-filtrado-colaborativo/>
- [70] Sarwar, B., Karypis, G., Konstan J. y Riedl, J. (s.f.) "Item-Based Collaborative Filtering Recommendation Algorithms" Obtenido de:
http://files.grouplens.org/papers/www10_sarwar.pdf
- [71] Jaadi, Z. (s.f.) "A Step-by-Step Explanation of Principal Component Analysis (PCA)"

Obtenido de: <https://builtin.com/data-science/step-step-explanation-principal-component-analysis>

[72] De Wu, J. (2021) “Análisis de Componentes Principales: Implementación en Python”
Obtenido de: <https://blog.damavis.com/analisis-de-componentes-principales-implementacion-en-python/>

[73] Kumar, R. (2018) “Understanding Principal Component Analysis” Obtenido de:
<https://medium.com/@aptrishu/understanding-principle-component-analysis-e32be0253ef0>

[74] (Scikit learn) “sklearn.model_selection.train_test_split” Obtenido de:
https://scikitlearn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[75] (InteractiveChaos) “MinMaxScaler” Obtenido de:
<https://interactivechaos.com/es/manual/tutorial-de-machine-learning/minmaxscaler>

[76] (FastAPI) “FastApi” Obtenido de: <https://fastapi.tiangolo.com/>

[77] Ramírez, S. “Using FastAPI to Build Python Web APIs” (s.f.) Obtenido de:
<https://realpython.com/fastapi-python-web-apis/>

[78] (Dio) “Dio” Obtenido de: <https://pub.dev/documentation/dio/latest/>

[79] (Android Studio) “Cómo ejecutar apps en Android Emulator” Obtenido de:
<https://developer.android.com/studio/run/emulator?hl=es-419>

[80] (Firebase) “Firebase Authentication” Obtenido de:
<https://firebase.google.com/docs/auth?hl=es-419>

[81] (Firebase) “Firebase Android Codelab- Cree un chat amigable” Obtenido de:
<https://firebase.google.com/codelabs/firebase-android?hl=es-419#0>

[82] (Becker, 2019) “Ya llegaron los codelabs a Android” Obtenido de: <https://developers-latam.googleblog.com/2019/01/yalllegaronloscodelabsdeandroid.html#:~:text=Un%20codelab%20es%20un%20instructivo,fueron%20pensados%20como%20capacitaciones%20presenciales>

[83] (Kaggle) “Kaggle” Obtenido de: <https://www.kaggle.com/>

ANEXOS

Enlace al boceto del diseño inicial del *frontend*

<https://www.figma.com/file/Sdf5J5OWfv0VXI7U762lq6/AdoptionMovement?type=whiteboard&node-id=0-1>

Enlace al repositorio de código en *GitHub*

<https://github.com/lucianunezalonso/tfg-22029926.git>

[PÁGINA INTENCIONADAMENTE EN BLANCO]