Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA DE SEGOVIA

Grado en Ingeniería Informática de Servicios y Aplicaciones

Diagnóstico de animales de compañía mediante AI

Alumna: Rebeca Caballero Suárez

Tutor/a/es: José Vicente Álvarez Bravo

Diagnóstico de animales de compañía mediante AI

Rebeca Caballero Suárez

Índice general

Lista de Imágenes

Lista de Tablas

Dedicado, con mucho cariño, a todos aquellos animales que han sido, son y serán miembros de una familia.

Agradecimientos

En eterna gratitud con la **Clínica Veterinaria San Lorenzo** y la **Clínica Veterinaria Nueva Segovia**, que han ofrecido todo su conocimiento, paciencia y cariño para llevar adelante este proyecto en pos de aquellas pequeñas criaturas que consideramos como un miembro familiar más.

También agradecimientos a mi familia y amigos, que siempre ofrecen un sitio donde poder recibir apoyo moral y dar un lugar de descanso para recuperar fuerzas, animando a superar cualquier obstáculo que me llegue a enfrentar.

Por último, y no menos importante, a mi querido tutor José Vicente Álvarez Bravo. Sin él, este trabajo nunca hubiese llegado a realizarse.

Comunicación Oficial de Inicio

Entrenamiento y desarrollo de una Inteligencia Artificial.

Capaz de detectar estados anómalos a partir de la introducción de datos sobre el estado actual del animal. Posteriormente al diagnóstico, sugerirá una serie de tratamientos, las más recomendables en el caso a tratar.

Se emplearán como base los conocimientos adquiridos sobre IA y Machine Learning para luego adaptarlas con la realización de un proyecto real.

El principal lenguaje de trabajo será Python, usando Jupyter Notebook como IDE.

Palabras claves: IA, AI, ML, Animal, Diagnóstico, Python, Jupyter Notebook.

Resumen

Este proyecto es motivado por el bienestar de nuestros queridos animales de compañía. En este caso se trata de **desarrollar modelos de IA** para el **diagnóstico animal o del paciente** con la idea de ofrecer una segunda opinión al veterinario, pudiendo ser posible agilizar su trabajo distribuyendo su concentración más ampliamente a otras necesidades.

Para ello, se requiso de un estudio e investigación del sector veterinario para identificar sus necesidades y ofrecer el mayor valor posible, además de la recogida de estos casos reales para su uso en el proyecto.

Una vez se determinó la prueba de referencia veterinaria y los distintos diagnósticos presuntivos contemplados, se procedió con el desarrollo y entrenamiento de varios algoritmos de aprendizaje supervisado en *Machine Learning*, separándolos por su objetivo:

- **Determinar la especie animal** sin previo indicativo. (***SpecAI***)
- **Predecir el diagnóstico presuntivo** del paciente con los datos facilitados de la prueba junto con la especie suministrada por la anterior IA. (***BlooDAI***)

Tras un análisis del desempeño de cada uno de los algoritmos, se determinaron aquellos mejores modelos finales a extraer, siendo incentivos de una elaboración de un programa o herramienta software futura.

Dentro del contenido de este proyecto se facilita un ejemplo de uso ideal dentro de una implementación software.

Abstract

This project is driven by the well-being of our beloved companion animals. In this case, the goal is to **develop AI models** for **animal or patient diagnosis**, with the aim of offering a second opinion to veterinarians. This could help streamline their work by allowing them to distribute their focus more broadly to other needs.

To achieve this, a study and analysis of the veterinary sector was required in order to identify its needs and deliver the highest possible value, along with the collection of real-world cases to be used in the project.

Once the veterinary reference test and the various presumptive diagnoses were determined, the development and training of several supervised *Machine Learning* algorithms was carried out, categorized by their objective:

- **To determine the animal species** without prior indication. (***SpecAI***)
- **To predict the patient's presumptive diagnosis** using the test data, along with the species inferred by the previous AI. (***BlooDAI***)

After analyzing the performance of each algorithm, the best-performing models were selected as candidates for future software tool development.

This project also includes an example of ideal usage within a software implementation.

Parte I

Memoria del Proyecto

Capítulo 1

Descripción del proyecto

Se tratará brevemente cómo se organizó la Memoria del Trabajo Fin de Grado (TFG).

1.1. Introducción

El "animal de compañía" es un concepto que se conoce desde tiempos del hombre prehistórico, según reflejan muchas teorías a raíz de la convivencia con el lobo-perro[1], conocido comúnmente como perro lobo[2]. Al igual que la tecnología, ha ido avanzando con el paso del tiempo hasta ser una comidilla habitual para determinar cuál de entre todos ellos es el mejor. Una competitivad amistosa que puede haber dentro de una comunidad de vecinos.

En el siglo XXI, se ha llegado a alcanzar una estabilidad de adopción y de abandono de estos nuevos miembros familiares hasta el punto de contar de su propio seguro obligatorio. O eso se pensaba hasta que surgió el COVID-19.

A raíz del confinamiento por el coronavirus SARS-CoV-2, no solo dio a pie al teletrabajo y a la exploración culinaria, sino también a saciar la soledad de algunos con la adopción de animales de compañía[32]. Se observó que para aquellos de los cuales no contaban de compañía humana en sus hogares, estos animales eran capaces de mejorar positivamente la psique de sus dueños, incluso es un hecho actual para los más jóvenes[26].

Aunque es cierto que tras la pandemia los casos de adopción y abandono volvieron a aumentar[21], fue un motivador clave de adopciones en esos momentos de tensión y miedo que azotaron a todos. Aquellas personas que anteriormente no lo tenían ni en consideración, ahora tienen la oportunidad de permitirse adoptar a uno de estos animales gracias a las consecuencias que surgieron del confinamiento mundial, entre ellos: el teletrabajo y el encariñamiento por éstos en tiempos de necesidad.

[1] Híbrido canino que resulta del apareamiento de un lobo gris y un perro.

Yo ya contaba con el cariño por los animales desde muy pequeña, y en 3º de primaria fue cuando se nos unió a la familia nuestra primera perrita, Trufa, de la cual nos encariñamos rápidamente. Como amante principal de los caninos, pensar en desarrollar algo que pueda ser beneficioso para ellos, quienes han formado parte de mi vida desde tan pequeña e incluso ahora, es algo con lo que me llenaría de orgullo y satisfacción, y más si lo sumamos con una tecnología que está en auge actualmente, la Inteligencia Artificial (IA)[31].

El gran éxito mundial de OpenAI tras su desarrollo de ChatGPT en 2022 ha impulsado la inversión en IA después de décadas en un falso estado ocioso[24], a pesar de las constantes críticas en sectores como en el arte[27] o en la educación[8]. Podemos destacar grandes empresas como Microsoft y Google con sus respectivas propuestas, Copilot y Gemini, quienes han llegado a invertir millones de dolares.

Teniendo en consideración a la mala usabilidad de esta tecnología o el impacto negativo que puede llegar a carrear en la sociedad humana como la conocemos, opino que si se les da un uso legítimo acorde al objetivo por el que se desarrolla incialmente, puede llegar a ser una gran facilitadora con cierto grado de veracidad para tareas complejas o de cierta incertidumbre, como podría ser: en tareas manuales de alto riesgo, en la metereología o, en este caso, en el diagnóstico de enfermedades.

1.2. Proceso de Investigación

Al tratarse sobre la salud de una especie, un ámbito muy complejo y con un historial muy variado, a continuación se detalla el proceso seguido para cumplir con el alcance y objetivos posteriormente definidos.

1.2.1. Colaboraciones

El mayor foco de este proyecto se haya en la colaboración directa con centros veterinarios, específicamente de la zona.

Estas colaboraciones permiten no solo al desarrollo de un proyecto con una funcionalidad real, sino a estrechar lazos entre el sector veterinario con la Universidad de Valladolid, al menos con el Campus María Zambrano.

A partir de estas, se adquieren conocimientos con mayor fiabilidad y mayor garantía de éxito para determinar qué datos son los que se analizarán y cómo se tratarán para desarrollar y entrenar la IA, no sin antes entender el mundo y la forma de trabajar del sector en cuestión.

La Clínica Veterinaria San Lorenzo tiene contacto directo con la **Asociación para la Defensa de los Animales de Segovia** y constan de maquinaria para realizar pruebas Hematológicas[2], Bioquímicas[3], Dermatológicas[4] y por Rayos X. Seguida a ella está la Clínica Veterinaria Nueva Segovia, con una instalación ligeramente más pequeña pero no menos profesional.

1.2.2. Conceptos Básicos

A continuación se detallan unos conceptos básicos en el estudio y diagnóstico de animales de compañía.

ANAMNESIS

La Anamnesis en el sector veterinario sería, en términos sencillos y bien resumido, el motivo de la consulta. Aunque parezca algo fácil de definir a primera vista, tiene un procedimiento de gran importancia para determinar el historial clínico del animal y los síntomas por los cuales se ha solicitado consulta.

Se trata de una interacción entre el médico y el dueño del paciente, y es común que se realicen las siguientes preguntas o de carácter similar:

- ¿Está al día con las vacunas?
- ¿Cuándo se realizó la última desparasitación?
- ¿Se encuentra en medicación?
- ¿Qué hábitos tiene su mascota?

Este proceso es esencial, y se realiza siempre antes de pasar a realizar cualquier prueba, ya que permite desarrollar un plan de atención personalizado y eficaz para su bienestar[6].

EXPLORACIÓN FÍSICA GENERAL (EFG)

Mientras que el dueño del paciente se encuentra respondiendo a las preguntas que el médico plantea, lo cual desencadenaría en la identificación de la anamnesis, se hace una comprobación general del estado físico del animal[35].

Habrá muy pocos casos en los que este proceso sea no realizado, ya sea por la claridad o motivo de la visita, y de la urgencia a ser tratado en el acto.

[2]Rama de la medicina que estudia la morfología de la sangre y los tejidos que la producen.

[3]Campo que se fundamenta de las actuaciones en las que, a través de diagnósticos, pruebas y fisiopatológica, ayuden al diagnóstico, pronóstico, tratamiento, seguimiento y prevención de la enfermedad.

[4]Disciplina que estudia y trata las enfermedades en la piel

La exploración incluye:

1. **Inspección general visual** del estado del animal.
2. **Auscultación pulmonar y cardíaca**. Se escuchan los sonidos naturales del organismo del paciente.
3. **Palpación**. Se determina la consistencia, movilidad, forma y tamaño de los órganos, además de detectar presencia de dolor. Sigue una orden sistemática durante el proceso.
4. **Percusión**. Se golpea levemente una zona para provocar un sonido audible y diferenciable. Normalmente en animales de gran tamaño.
5. **Toma de constantes vitales**. Principalmente son medidas la temperatura, frecuencia cardíaca y respiratoria, y el pulso.

Figura 1.1: Exploración Física General en gatos [4]

La anterior figura es una referencia de una exploración física general para felinos *(Figura 1.1)*.

1.2.3. Casos Reales

Durante el intervalo desde finales de Mayo y a finales de Junio, se ha ido recopilando manualmente las consultas realizadas en la Clínica Veterinaria San Lorenzo y así, además de familiarizarme con la herramienta de Software a nivel de gestión veterinaria **WinVet**, conocer de primera mano los procedimientos seguidos antes del diagnóstico.

De esta manera, se pueden observar los casos más comunes y aquellos que sean más idóneos con los que trabajar y desarrollar una funcionalidad, no solo a nivel de programación sino también para el principal beneficiario.

Un total de 120 casos fueron recogidos en las clínicas veterinarias, donde se excluyeron datos sensibles tanto del cliente animal como el de su dueño principal. Estos datos se trataron posteriormente para una mayor nitidez, del cual poder realizar un estudio concienzudo y detallado.

En ellos se reflejan:

- **Especie** del Animal de compañía
- **Raza**
- **Género**
- **Edad**
- **Peso**
- **Anamnesis** o motivo de la visita
- **Pruebas** realizadas en la consulta
- **Diagnóstico** emitido[5]
- **Tratamientos** recetados[6]

Almacenados en Excel, se desarrolló un listado de especies, razas, y anamnesis más comunes junto con una clasificación de todas las pruebas dadas en esas consultas y cuánto porcentualmente fueron realizadas en su totalidad.

[5]Es muy importante saber que el diagnóstico, tanto para animales como para humanos, casi siempre será **presuntivo**. Raramente se tendrán *diagnósticos diferenciales* en un sector tan ambiguo y afectado por tantos condicionales como es la salud.

[6]A su vez, los tratamientos dependerán del historial médico actual del paciente y de sus condiciones físicas en el momento que se realice la consulta.

1.2.4. Investigación

En el estudio podemos destacar, sin sorpresa, 2 especies: **Caninos y Felinos**, siendo el primer mencionado el más habitual con un 77 % de los casos. Aunque se dote de una tercera especie, el Conejo, no son casos habituales entre ambas clínicas *(Figura 1.2)*.

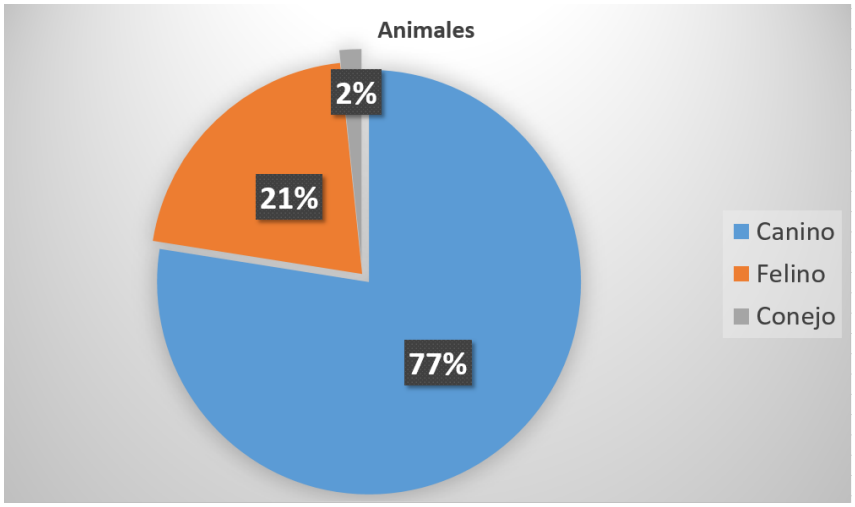

Figura 1.2: Animales o especies identificadas

Los datos adquiridos son muy significativos para saber con quiénes sería ideal poner el foco de atención, al contrario que en el género. No hay predilección entre un macho o una hembra *(Figura 1.3)*.

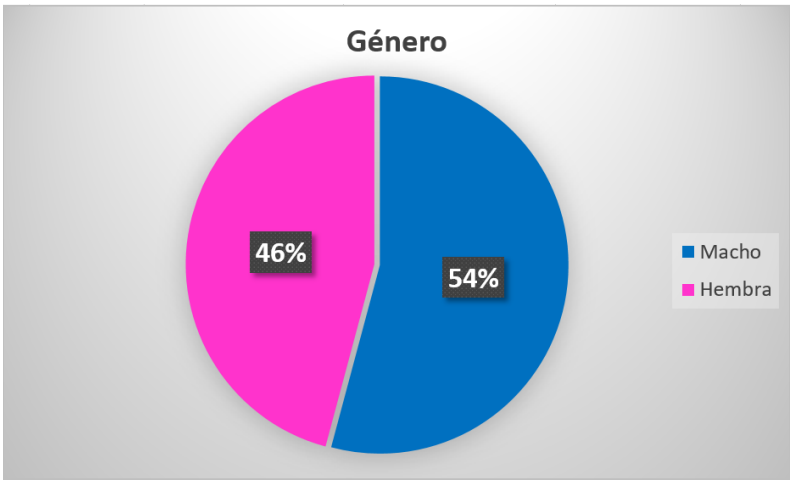

Figura 1.3: Género de los animales domésticos

Además, podemos destacar las razas más predilectas según felinos y caninos, que serían el **Europeo** *(Figura 1.4)*, y **Mestizo** seguido del **Labrador Retriever** *(Figura 1.5)*.

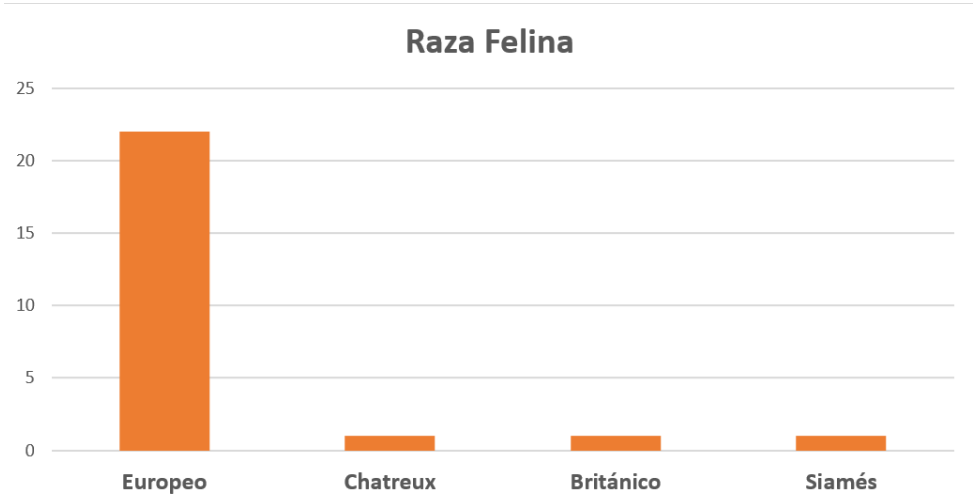

Figura 1.4: Razas felinas

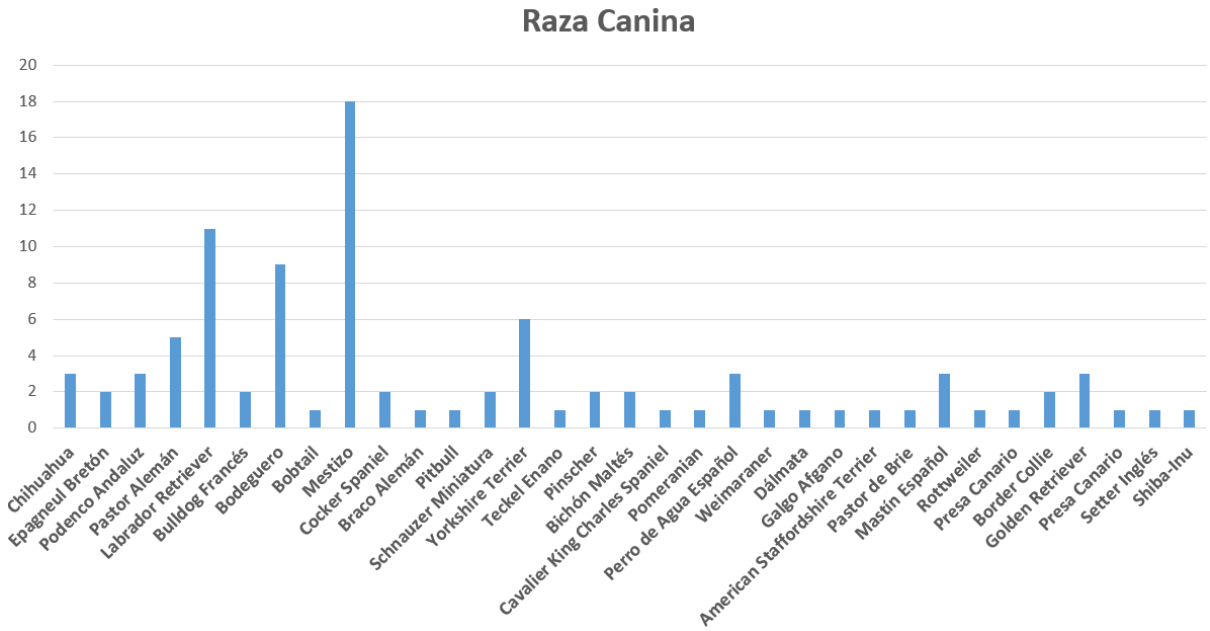

Figura 1.5: Razas caninas

Adicionalmente, se extrajo la volumetría de los motivos o anamnesis por el que un paciente es llevado a consultoría a partir de una gráfica más visual e intuitiva, de la cual podamos comprender de un vistazo la inquietud de los interesados *(Figura 1.6)*.

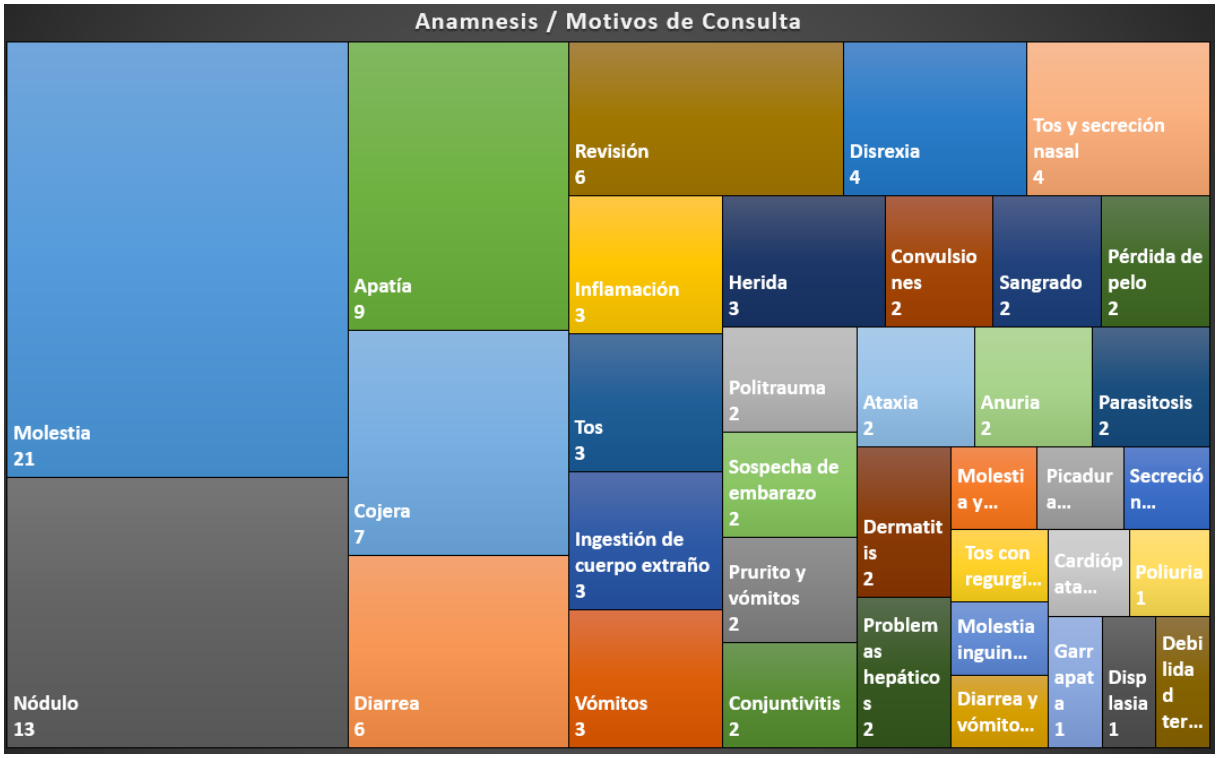

Figura 1.6: Anamnesis o motivo para consulta

Aunque no se llegue a apreciar, se han diferenciado de entre las 120 consultas recopiladas un total de **36 tipos de anamnesis**.

Las causas más comunes de ir al veterinario son el estado de **molestia** del animal o la presencia de **nódulos** o bultos extraños. Ésto último suele ser de gran preocupación para el dueño al ser un posible síntoma de tumor, ya sea benigno o maligno.

La **apatía** del animal también es una casuística muy común, junto con la inapetencia o falta de hambre.

Y finalmente, podemos detectar cuántas de las consultas recogidas dieron un diagnóstico positivo o negativo, asimismo de saber el más común en la temporada de Mayo-Junio.

En este intervalo sería por **Cuerpo Extraño**, usualmente por una espiga clavada en la oreja o en la nariz del animal *(Figura 1.7)*. Esto puede ser debido por la temporada en la que se recogieron estas muestras.

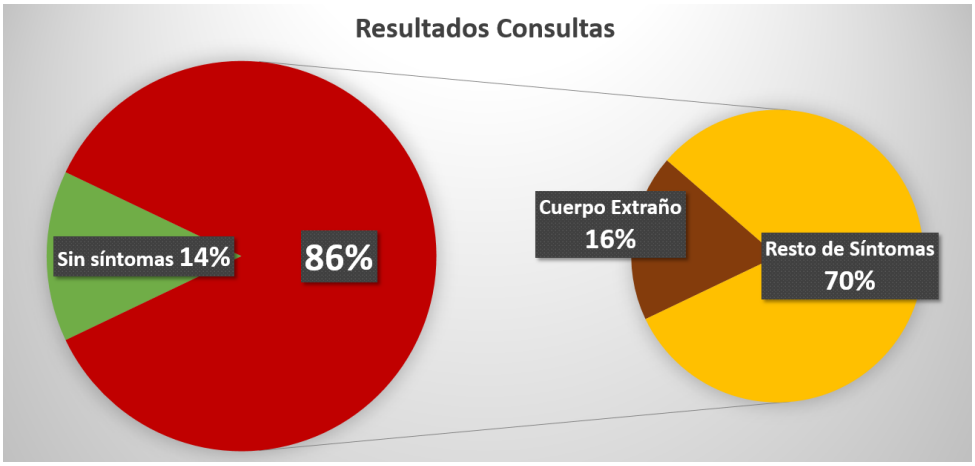

Figura 1.7: Resultados y diagnóstico más común

Aunque todos los datos extraidos anteriormente son importantes, hay que definir previamente el método o la **prueba diagnóstica** por el que centrarse. No debería únicamente seleccionarse por su sencillez o fácil interpretación de los datos que llega a recoger, sino también por su usabilidad real.

Por lo tanto, hay que analizar todas las pruebas realizadas entre las 120 consultas e identificar cuál de entre ellas sería el punto de partida para el **proyecto DACAI**.

Para ello, se desarrolló una especie de **checklist de pruebas** en el Excel de consultas *(Figura 1.8)*. De esta forma, se refleja en otra tabla en el que recoge las veces aplicadas para transformarlas en un valor porcentual *(Figura 1.9)*.

							PRUEBAS								
EFG	Hemograma	Bioquímica	Frotis Sanguíneo	Urianálisis	Electrolitos	Dermatología	Citología	Incisión	Test Leishamnia	Test fluoresceína	Exp. Otoscopio	Traumatología	Ecografía	Radiografía	Sedación
☐	☐	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐
☑	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑
☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
☐	☐	☐	☐	☐	☐	☐	☑	☑	☐	☐	☐	☐	☐	☐	☐
☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
☑	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑	☐

Figura 1.8: Listado de pruebas

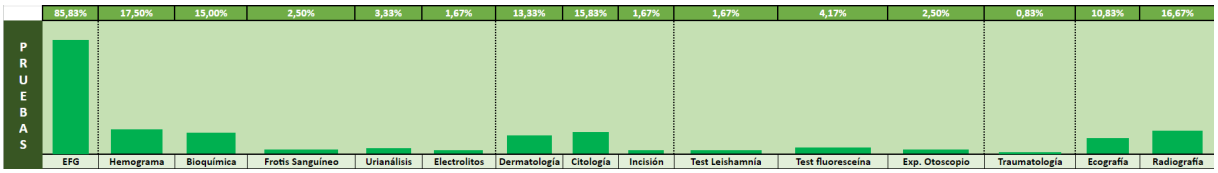

Figura 1.9: Tabla porcentual de las pruebas

Tras esto, podemos confirmar que aproximadamente en el 86 % de los casos se realiza una Exploración Física General. Como se mencionó anteriormente, esta prueba es prácticamente llevada a cabo casi siempre en todas las consultas veterinarias.

Descartándola, existen **3 campos** distintos de pruebas diagnósticas que predominan sobre el resto, pero son muy parejas entre sí:

- Análisis de sangre.
- Enfermedades en la piel.
- Rayos X.

1.2.5. Conclusiones

Podemos resumir toda esta investigación en lo siguiente:

- La **población canina** como miembro familiar destaca sobre el resto, seguida de los **felinos**.
- El género no es un factor predominante.
- A excepción de los felinos, hay gran variedad de razas caninas a considerar en cada consulta y diagnóstico.
- Los motivos de consulta pueden ser muy variados y desencadenarse en distintos tipos de diagnósticos según la fecha de la visita, las pruebas realizadas y las condiciones del animal.
- Y por último pero no menos importante, el **Hemograma**, la **Citología** y la **Radiografía** son las pruebas más habituales en el día a día de una clínica veterinaria profesional.

Por lo tanto, se tomaron las siguientes **decisiones** para el desarrollo del proyecto:

1. **Caninos y Felinos:** Estas dos especies serán nuestros principales pacientes a tratar, y del cual gestionar los posibles diagnósticos a partir de la prueba escogida con la que trabajar.

2. **Hemograma:** Elegida a partir de ser la prueba más comúnmente realizada en el ámbito veterinario. Se barajeó también incluir la **Bioquímica** al ser normalmente realizada junto con el hemograma, y se dota de una investigación previa. Sin embargo, se optó por la efectividad y exactitud de las IAs a desarrollar sobre un gran rango de alcance que podría acarrear una investigación "ociosa".

1.3. Objetivos del trabajo

A partir de estos datos recopilados y analizados, identificamos el alcance y los siguientes objetivos como **base del proyecto DACAI**, afectando así a lo estimado inicialmente.

1. A nivel práctico:
   - **Desarrollar y entrenar la IA que hará un diagnóstico al paciente.** Restringida únicamente por la población de datos de prueba y su capacidad de diagnosticarle sólo mediante el **Hemograma**.
   - Las IAs ha desarrollar deberán tener un **porcentaje de acierto equivalente** al modelo de entrenamiento predominante y al volumen de datos facilitado en sus diagnósticos presuntivos.
   - **SpecAI**, nombre de una de las IAs desarrolladas, se encargará de **identificar a la especie** en cuestión a partir de los datos de un hemograma.
   - **BlooDAI**, nombre principal de la IA centrada para casos por análisis de sangre, deberá ser capaz de recopilar y tratar todos los **datos de un hemograma completo real** junto con lo identificado por SpecAI para **determinar un diagnóstico presuntivo**.
   - **Extraibles y portables** para una implementación futura en una herramienta software.
2. A nivel teórico:
   - Tomar el rol de co-examinador durante el **aprendizaje** de un futuro veterinario o auxiliar para el diagnóstico de enfermedades por prueba Hematológica, sugiriendo una **segunda opinión**.
   - Y por ende, **agilizar y facilitar el trabajo** de un veterinario profesional gracias a su fiabilidad al determinar un diagnóstico.
3. Y a nivel personal:
   - Ser el primer **punto de conexión** para el departamento de informática en el Campus María Zambrano con Clínicas Veterinarias de la zona.
   - **Fomentar trabajos relacionados** con el sector veterinario.

1.4. Estado del Arte

Hay una gran cantidad de estudios y artículos sobre las posibles aplicaciones de la IA y las ayudas que aportaría no solo al veterinario sino también para los pacientes y sus dueños. Sin embargo, apenas existen ejemplos que realmente se hayan llegado a desarrollar y alcanzado a los profesionales sin un alto coste.

Una mención sería **PicoxIA**, la primera herramienta software para radiográfias en el sector veterinario. Mediante la colaboración profunda con radiólogos, está especializado en zonas del abdomen, tórax y displasia de cadera accesible desde la nube, cuyo objetivo de la IA es realizar un diagnóstico completo a nivel de **traumatología**. Además, ella consta de publicaciones sobre sus actuaciones en revistas revisadas por pares.

La más destacable con nuestra temática, sin embargo, sería **Vetscan IMAGYST** (*Figura 1.10*). A partir del escaneo de la frotis de sangre del animal junto con los resultados obtenidos del hemograma, entre otros tipos de pruebas, generará un **estudio exhaustivo** del caso del cual se pueda almacenar en el registro del cliente o ser enviado a un patólogo. Se trata de la aplicación más completa hasta ahora, pero tiene una gran restricción: toda la gestión veterinaria debe previamente estar en **Vetscan FUSE** para aprovechar todo su potencial.

Figura 1.10: Vetscan IMAGYST

Es cierto que el Proyecto **DACAI** no consta de una herramienta o interfaz para poder tener un uso directo con los interesados, pero sí tiene **2 objetivos** que el resto de mencionados no tienen: **determinar la especie** a partir de los datos hemográficos facilitados sin un indicativo previo y que a su vez **facilite el diagnóstico presuntivo** del paciente en cuestión junto con dicha prueba.

	Hematología	Detección de Especie	Diagnóstico Presuntivo	Análisis Completo
PicoxAI	×	×	✓	✓
Vetscan	✓	×	×	✓
DACAI	✓	✓	✓	×

Cuadro 1.1: Estado del Arte: PicoxIA, Vetscan y DACAI

Valorando las labores de cada una de las IAs utilizadas en las herramientas de software mencionadas, obtenemos las anteriores conclusiones comparándolo con las determinadas en este proyecto (*Cuadro 1.1*).

Capítulo 2

Metodología de Trabajo

A continuación se detallará la metodología seguida para el desarrollo del proyecto junto con las herramientas utilizadas para su seguimiento.

2.1. Proceso de desarrollo: Agile - Scrum & Kanban

A raíz de la naturaleza del proyecto donde se destaca por su gran incertidumbre y desconocimiento previo, se ha seguido un proceso de trabajo en el que se distinga por su flexibilidad y en el que se ofrezca constante valor sin necesidad de ser algo material o "funcional", como la adquisición de conocimientos veterinarios. Por lo tanto, estamos hablando de una **Metodología Ágil (*Agile*)**.

En un marco ágil se colabora directamente con los interesados y usuarios finales, conocidos también como parte del conjunto de *stakeholders*, para poder **definir requisitos** dentro de un rango temporal para **entregar valor al producto de manera iterativa**. Esto permite tener **mejor respuesta ante el cambio**, muy esencial para cuando todos los implicados no saben lo que realmente necesitan de lo que quieren y se necesite además un traspaso de conocimientos (*Knowledge Transfer - KT*).

Esta base viene facilitada por el **Manifiesto Ágil**[3] definida por hasta 17 expertos en desarrollo de software.

Junto con ésto, se ha usado la filosofía de **Scrum**[22] para trabajar de manera iterativa y entregar valor frecuentemente mediante ***Sprints***[1] a partir de una colección de tareas a realizar en el proyecto, conocida como ***Backlog*** *(Figura 2.1)*, sumada a una gestión visual del trabajo ofrecida por el marco **Kanban**[25] *(Figura 2.2)*.

[1]Períodos de tiempo en el que se tienen definidos el trabajo ha desarrollar, conocido como *Sprint Backlog*. Suelen tener un alcance ideal entre 3-4 semanas como máximo.

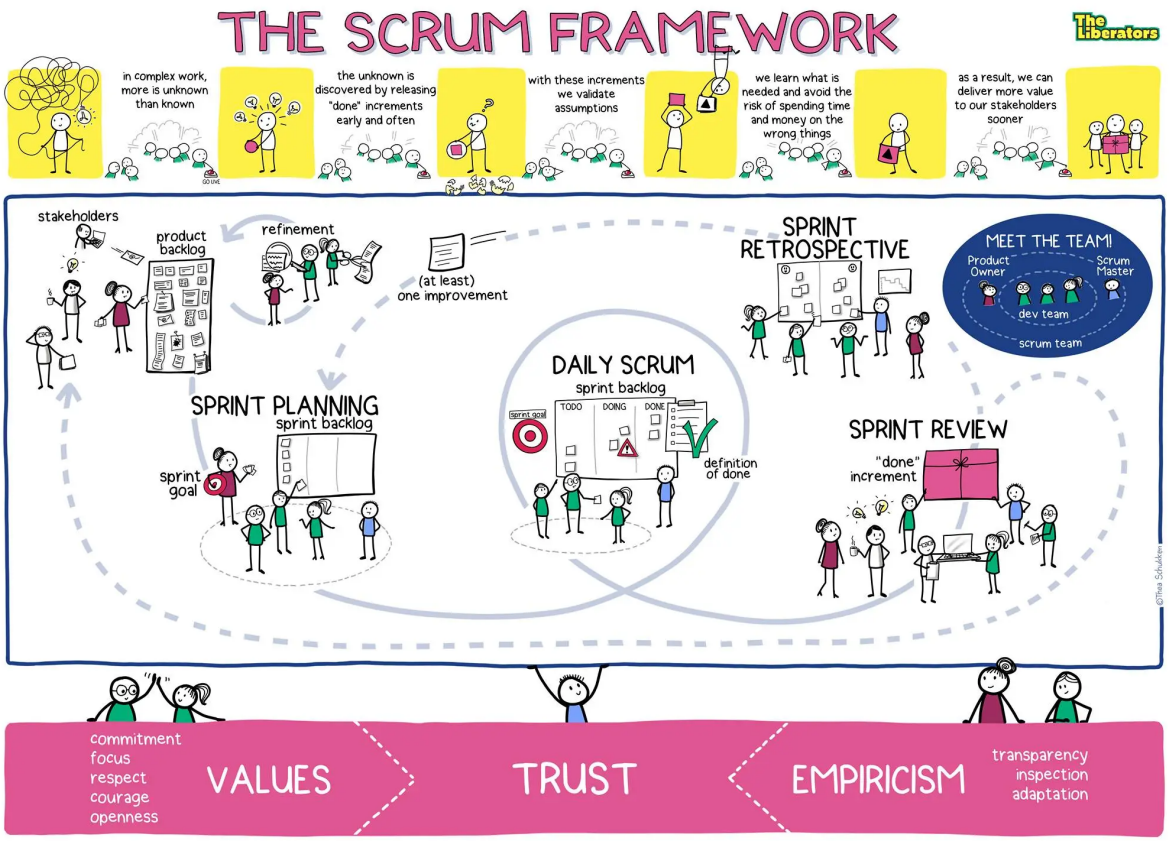

Figura 2.1: Metodología Ágil: Scrum

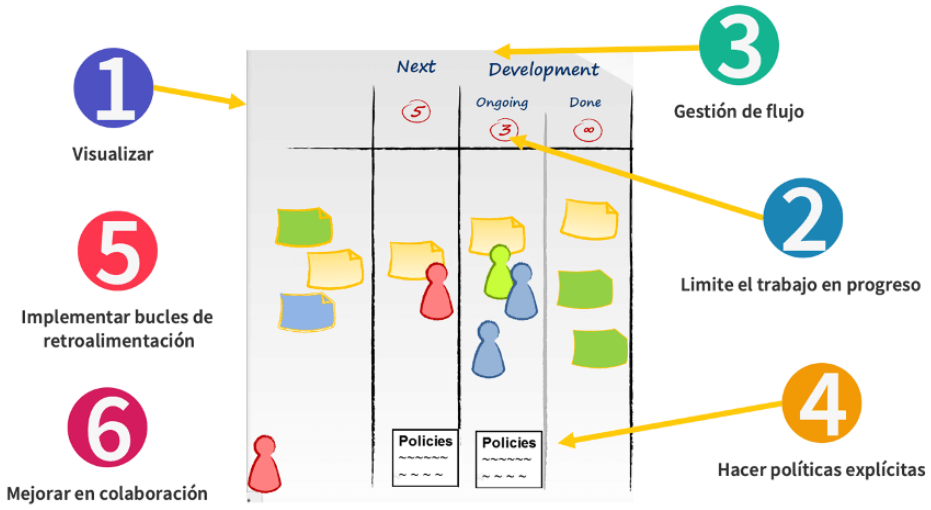

Figura 2.2: Metodología Ágil: Kanban

2.2. JIRA: Gestión de Proyectos

Aunque hay muchas herramientas que puedan hacer una gestión y un seguimiento óptimos en proyectos software, **JIRA** es líder en el mercado empresarial. Ésta permite **planificar, rastrear y gestionar proyectos** de manera eficiente a partir de una **personalización sencilla e intuitiva** al proyecto en cuestión, pudiendo no sólo trabajar con metodologías ágiles sino también tradicionales, como "En Cascada" (*Waterfall*).

Desarrollada por **Atlassian**, está **en constante mejoría** y consta de una gran **versatilidad de integración** a otras herramientas, como Microsoft Teams o BitBucket.

Otro de sus grandes desarrollos sería **Confluence**, que facilita la colaboración, comunicación e información que tiene el equipo a partir de páginas de contenido en línea ya sea asociadas o no a un proyecto creado en JIRA.

2.2.1. Definición de Tareas: Backlog

Antes de empezar con un proyecto, siempre hay que analizar los pasos necesarios a realizar para llevarla acabo. Por lo tanto, habría que definir las tareas y su tipo.

Los tipos de tareas (o incidencias/tickets en aspectos técnicos de JIRA) aparecieron a partir de las metodologías ágiles, en las que diferenciaron estos procesos a llevar acabo, en pos de mantener un seguimiento sostenible, principalmente en:

- **Historias de Usuario (HU):** Normalmente, son aquellas en las que se definen características o funcionalidades desde el interés propio de un cliente o usuario final. Por lo tanto, deben ser descritas de una manera entendible para ellos junto con todo el análisis técnico del equipo de desarrollo. Éstas se estimarán para poder gestionarlas en *Sprints*.

- **Tareas Técnicas:** Se tratan de aquellas acciones requeridas para poder completar una HU en un aspecto interno del desarrollo, como podría ser crear una base de datos, configuraciones previas...

- **Épicas:** Historias de usuarios que son demasiado grandes al tener un alcance extenso. Por ejemplo, la integración de algún software implica varias funcionalidades para llevarse acabo y, por ende, en un pequeño intervalo de tiempo (como un *Sprint*) por sí solo no podría realizarse. Por lo tanto, son el origen de otras historias más pequeñas.

Se podrán observar estos tipos en la *(Figura 2.3)*.

Jira

Proyecto: DACAI
Ordenado por: Clave ascendente, entonces Creada descendente
1–70 de **70** como en: **02/mar/25 6:33 PM**

T	Clave	Resumen	Persona asignada	Estado	Resolución	Creada	Actualizada	Sprint
	DACAI-1	Clínica Veterinaria San Lorenzo 1.0	Rebeca Caballero Suárez	FINALIZADA	Listo	09/jul/24	07/nov/24	Colaboraciones & Metodologías
	DACAI-2	Clínica Veterinaria Nueva Segovia 1.0	Rebeca Caballero Suárez	FINALIZADA	Listo	09/jul/24	07/nov/24	Colaboraciones & Metodologías
	DACAI-3	Clínica Veterinaria CVS	Rebeca Caballero Suárez	CANCELADO	Listo	09/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-4	Clínica Veterinaria WE CAN	Rebeca Caballero Suárez	CANCELADO	Listo	09/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-5	Hospital Clínico Veterinario UPM	Rebeca Caballero Suárez	CANCELADO	Listo	09/jul/24	07/nov/24	Colaboraciones & Metodologías, Investigación & Memoria 1.0
	DACAI-6	DACAI-1 Solicitud de Colaboración	Rebeca Caballero Suárez	FINALIZADA	Listo	09/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-7	DACAI-1 Inicio: Objetivos TFG	Rebeca Caballero Suárez	FINALIZADA	Listo	09/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-8	DACAI-1 Investigación de Diagnósticos y Metodología de trabajo	Rebeca Caballero Suárez	FINALIZADA	Listo	09/jul/24	17/jul/24	Colaboraciones & Metodologías
	DACAI-9	DACAI-2 Solicitud de Colaboración	Rebeca Caballero Suárez	FINALIZADA	Listo	10/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-10	Colaboraciones	Rebeca Caballero Suárez	VALIDANDO	*Sin resolver*	10/jul/24	17/jul/24	
	DACAI-11	DACAI-2 Inicio de Objetivos TFG	Rebeca Caballero Suárez	FINALIZADA	Listo	12/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-12	DACAI-2 Investigación: Diagnósticos y Metodología de trabajo	Rebeca Caballero Suárez	FINALIZADA	Listo	12/jul/24	17/jul/24	Colaboraciones & Metodologías
	DACAI-13	DACAI-3 Solicitud de Colaboración	Rebeca Caballero Suárez	FINALIZADA	Listo	12/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-14	DACAI-3 Inicio de Objetivos TFG	Rebeca Caballero Suárez	CANCELADO	Listo	12/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-15	DACAI-4 Solicitud de Colaboración	Rebeca Caballero Suárez	FINALIZADA	Listo	12/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-16	DACAI-4 Inicio de Objetivos TFG	Rebeca Caballero Suárez	CANCELADO	Listo	12/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-17	DACAI-5 Solicitud de Colaboración	Rebeca Caballero Suárez	CANCELADO	Listo	12/jul/24	05/ago/24	Colaboraciones & Metodologías, Investigación & Memoria 1.0
	DACAI-18	Herramientas de Seguimiento: Mural + JIRA	Rebeca Caballero Suárez	FINALIZADA	Listo	12/jul/24	12/jul/24	Colaboraciones & Metodologías
	DACAI-19	Reunión (Planning) 14/05/24: Arranque del TFG	Rebeca Caballero Suárez	FINALIZADA	Listo	12/jul/24	17/jul/24	Colaboraciones & Metodologías

Figura 2.3: Ejemplo de Tareas/HUs/Épicas

Como se ha mencionado anteriormente, las HUs se deberán de **estimar** para poder ser parte del *Sprint Backlog* correspondiente. Esta estimación comúnmente se suele hacer no por horas sino por **Puntos de Historia (PH)**, que representan el esfuerzo y la complejidad del desarrollo en cuestión.

En este caso, se ha utilizado la **sucesión de Fibonacci** para representar no solo su complejidad, sino también su nivel de incertidumbre *(Figura 2.4)*.

Figura 2.4: Definición de Puntos de Historia seguido

Una vez definidas, se podrán gestionar para ofrecer valor constante mediante iteraciones de desarrollo, en este caso ***Sprints*** *(Figura 2.5)*. Esto luego se podrá visualizar temporalmente gracias al **Cronograma** *(Figura 2.6)*.

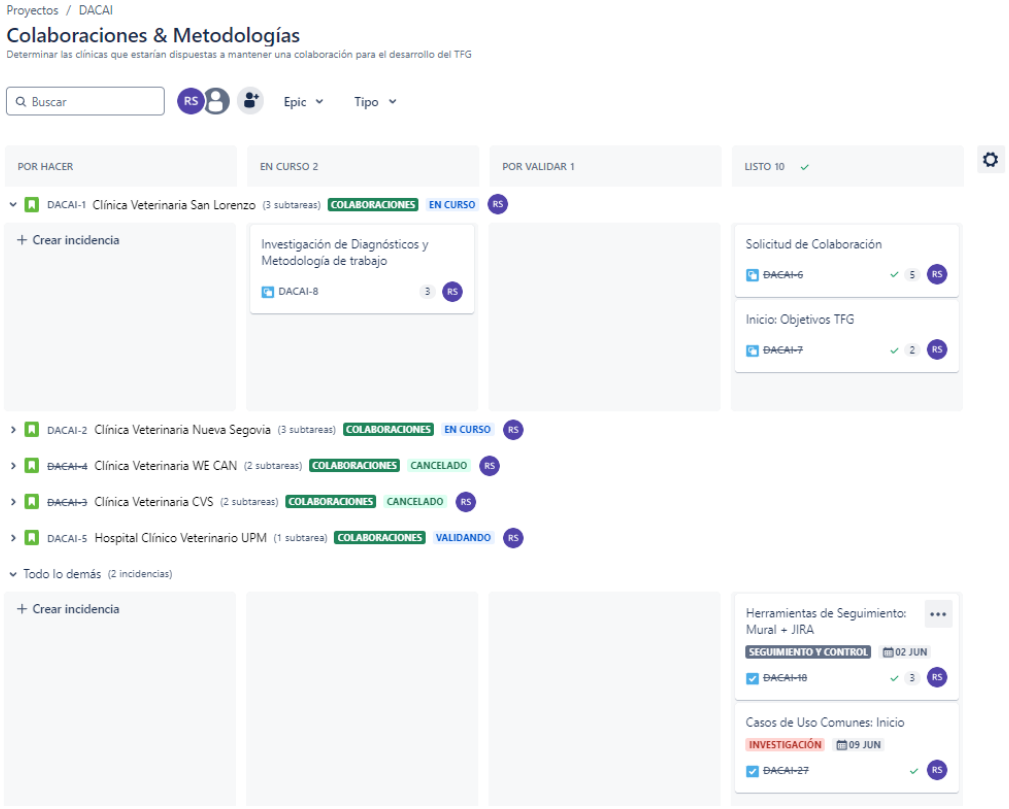

Figura 2.5: Ejemplo de Sprint

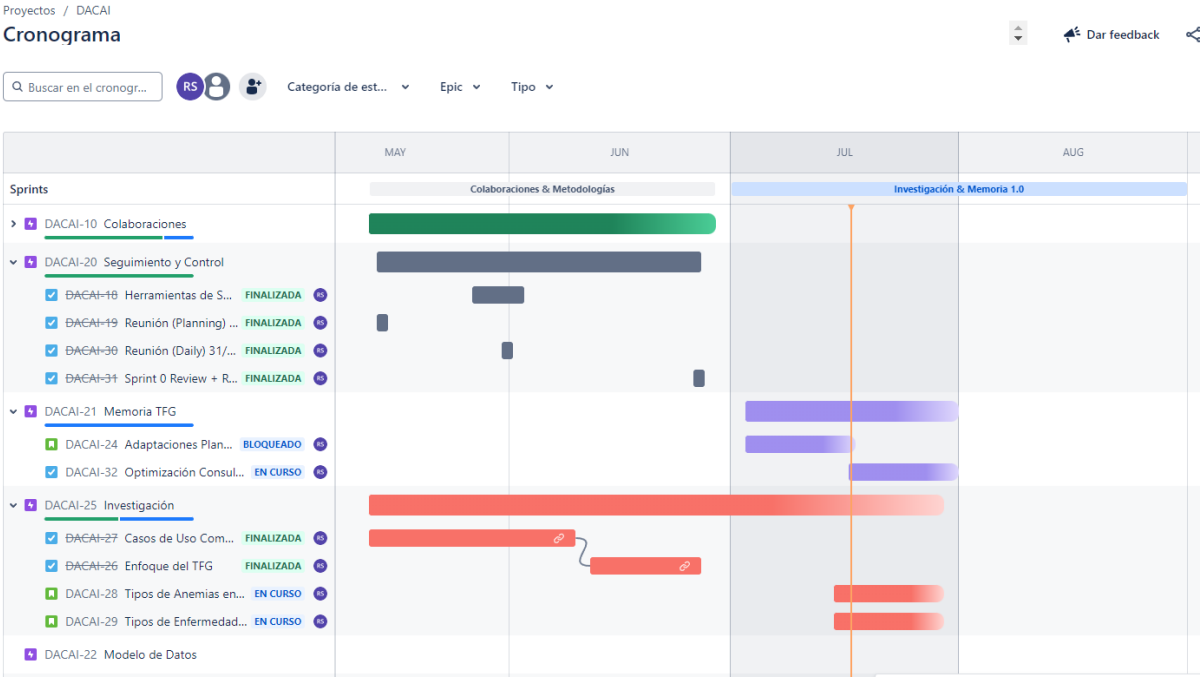

Figura 2.6: Ejemplo de Cronograma

Si se requiriese, podremos hacer un filtrado para extraer la información de un cierto rango o tipo de tareas en cualquier momento en el buscador de incidencias *(Figura 2.7)*.

Jira
Proyecto: DACAI
Estado: Finalizada
Ordenado por: Creada descendente
1–22 de **22** como en: **28/ago/24 4:46 PM**

T	Clave	Resumen	Persona asignada	Informador	Pr	Estado	Resolución	Creada	Actualizada	Fecha de vencimiento
	DACAI-38	Gráficas de Consultas 1.0	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/ago/24	18/ago/24	
	DACAI-36	DACAI-29 Búsqueda de Información y/o Artículos	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	05/ago/24	28/ago/24	
	DACAI-34	DACAI-28 Búsqueda de Información y/o Artículos	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	05/ago/24	05/ago/24	
	DACAI-33	Reunión (Daily) 17/07/24: Sincronización	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	17/jul/24	26/jul/24	17/jul/24
	DACAI-32	Optimización Consultas 1.0	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	17/jul/24	28/ago/24	
	DACAI-31	Sprint 0 Review + Retrospective + Planning 25-26/06/24	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	17/jul/24	17/jul/24	26/jun/24
	DACAI-30	Reunión (Daily) 31/05/24: Correo Complutense	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	17/jul/24	17/jul/24	31/may/24
	DACAI-27	Casos de Uso Comunes: Inicio	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	17/jul/24	09/jun/24
	DACAI-26	Enfoque del TFG	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	05/ago/24	26/jun/24
	DACAI-24	Adaptaciones Plantilla UVa	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	18/ago/24	
	DACAI-19	Reunión (Planning) 14/05/24: Arranque del TFG	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	17/jul/24	14/may/24
	DACAI-18	Herramientas de Seguimiento: Mural + JIRA	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	12/jul/24	02/jun/24
	DACAI-15	DACAI-4 Solicitud de Colaboración	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	12/jul/24	
	DACAI-13	DACAI-3 Solicitud de Colaboración	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	12/jul/24	
	DACAI-12	DACAI-2 Investigación: Diagnósticos y Metodología de trabajo	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	17/jul/24	
	DACAI-11	DACAI-2 Inicio de Objetivos TFG	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	12/jul/24	12/jul/24	
	DACAI-9	DACAI-2 Solicitud de Colaboración	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	10/jul/24	12/jul/24	
	DACAI-8	DACAI-1 Investigación de Diagnósticos y Metodología de trabajo	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	09/jul/24	17/jul/24	
	DACAI-7	DACAI-1 Inicio: Objetivos TFG	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	09/jul/24	12/jul/24	
	DACAI-6	DACAI-1 Solicitud de Colaboración	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	09/jul/24	12/jul/24	
	DACAI-2	Clínica Veterinaria Nueva Segovia 1.0	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	09/jul/24	17/jul/24	28/jun/24
	DACAI-1	Clínica Veterinaria San Lorenzo 1.0	Rebeca Caballero Suárez	Rebeca Caballero Suárez	=	FINALIZADA	Listo	09/jul/24	17/jul/24	25/jun/24

Figura 2.7: Visualización como lista de tareas finalizadas

El caso que se aplica es de extraer **aquellas finalizadas entre el 17 Julio y 18 Agosto**.

2.3. Herramientas de Desarrollo

Se ha trabajado con datos registrados en Excel y transformados en un fichero CSV para poder usarlo durante el entrenamiento y desarrollo de las IAs.

Estas IAs se hicieron con lenguaje **Python**, específicamente de la versión **3.11.5**, junto con las siguientes especificaciones básicas:

- **Conda**: 23.7.4
- **Anaconda Navigator**: 2.5.0
- **Jupyter Notebook**: 6.5.4

Anaconda Navigator es una interfaz gráfica de usuario (GUI) diseñada para facilitarnos, sobre todo, la gestión de todos los paquetes en **Python**, o R si se quisiese. Además, nos **permite crear y manejar entornos virtuales** si buscamos un uso más específico o tener un entorno de seguridad para desarrollos más actualizados y no "romper" otros pendientes de optimizar sin recurrir a línea de comandos. En resumen, es ideal

para aquellos que sean **principiantes** y busquen una forma de trabajar más visual.

Jupyter Notebook es una herramienta también interactiva que permite **almacenar documentos** que contengan código, ecuaciones, visualizaciones y texto narrativo. Es comúnmente usado para análisis de datos, aprendizaje automático (como *Machine Learning*), y exploración científica por su gran flexibidad.

En este caso, se ha usado Jupyter Notebook para **facilitar la documentación técnica del proyecto** a cualquier usuario nuevo, enfocando esta memoria como un punto de inicio para el proyecto.

Capítulo 3

Presupuesto

En este capítulo se abordará el coste que ha sido este proyecto si lo llevásemos a un ámbito real.

3.1. Estimación del Esfuerzo

En este caso, al seguir un marco de trabajo ágil, no se conciencia a trabajar con estimadores cuantificables como el tiempo que nos permitan facilitar el estudio económico. Sin embargo, esto se ve beneficiado para **adaptaciones** a causa de cambios ya sea externos o internos, **trabajando con un margen** para asegurar siempre la **calidad antes que lo acordado** *(Figura 3.1)*.

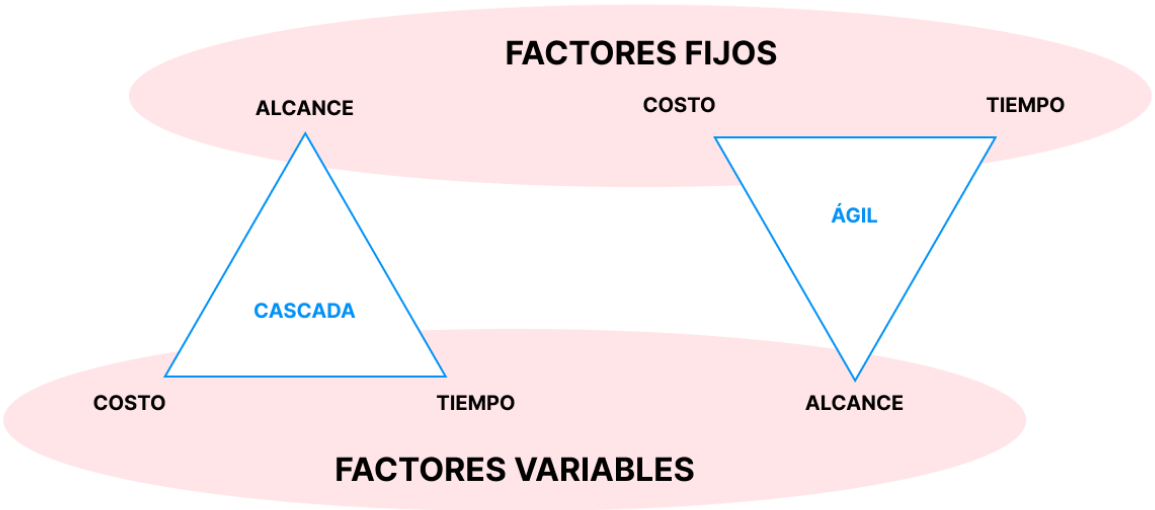

Figura 3.1: Triángulo de Hierro: Cascada VS Ágil

Por lo tanto, como hemos mencionado anteriormente, hemos trabajado con una estimación por **Puntos de Historia** (PH). Y todo el proyecto ha sumado un total de **138PH**[1].

3.2. Planificación Temporal

Para estimarlo a un rango inferior de trabajo, es decir, por meses, gracias a mi experiencia laboral lo más común es que una persona pueda abarcar de media **16PH/mes**. De tal manera, el tiempo saldrá de entorno 8 meses y medio. El momento **ideal** de trabajo sería de **inicios de año hasta finales** de éste que incluirá vacaciones de verano, pudiendo estimar finalmente de una **duración total del proyecto de 9 meses**.

En nuestro caso, se inició a mediados de Mayo, lo que hace que no cumpla con el ideal al tener también vacaciones de navidad *(Figura 3.2)*.

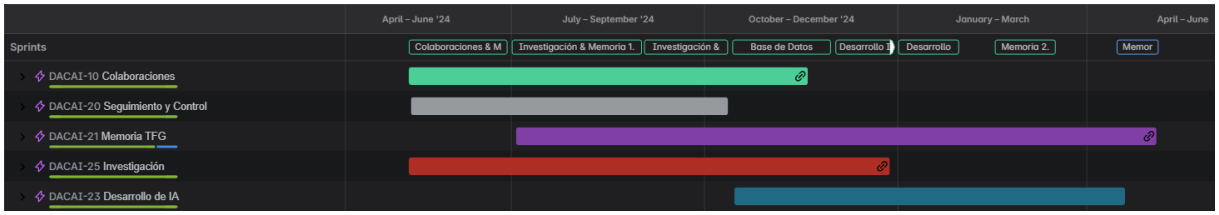

Figura 3.2: Cronograma final del TFG real

3.3. Presupuesto Económico

Para la estimación del presupuesto económico que se ha realizado en este proyecto, únicamente se contará de los recursos humanos en un ámbito real. Esto es porque se ha trabajado con herramientas libres u *Open Source*, accesible para cualquiera. Sólo existirán limitaciones de rendimiento del hardware con el que el trabajador desee trabajar.

3.3.1. Recursos Humanos

El trabajo realizado se reflejará entre 2 perfiles autónomos: **Analista de Datos y Desarrollador de Python** Estos perfiles se considerarán como **Junior** por los conocimientos y experiencia que contamos.

[1]Estos PHs están adaptados al trabajador que realiza la acción, en este caso en mi nombre. Los PHs son acordados por todos los implicados en el desarrollo del proyecto, no siendo equivalente entre otros equipos

Habrá que tener en cuenta también la diferencia entre **sueldo Neto y Bruto**, por lo que estimaremos[2] lo siguiente:

- **IRPF**[20]: 10 %
- **Cuota de Autónomos**: 80 €/mes

Y el tiempo de trabajo de media realizado es de 4h/día, haciendo que:

- **Mes**: 80h total, **40h** equitativamente.
- **Proyecto**: 720h total, **360h** equitativamente.

En el caso del **Analista de Datos**, tendríamos que tener en cuenta las **tarifas** *(Cuadro 3.1)* con las que estimar:

	Junior	**Intermedio**	**Senior**
Tarifa Mínima €/h	14 €	28 €	55 €
Tarifa Máxima €/h	23 €	46 €	+92 €

Cuadro 3.1: Tarifas de Analista de Datos

A partir de la tarifa media y el tiempo de trabajo mencionado anteriormente, podremos estimar el **coste total como analista** *(Cuadro 3.2)*.

	Coste
Tarifa Bruta Media por Hora	19 €/h
Total Bruto	6840 €
Neto por Hora	~15,1 €/h
Total Neto	~5436 €

Cuadro 3.2: Total del Analista de Datos

En el otro caso, del **Desarrollador de Python**, tendríamos que tener en cuenta estas otras **tarifas** *(Cuadro 3.3)*:

	Junior	**Intermedio**	**Senior**
Tarifa Mínima €/h	18 €	37 €	74 €
Tarifa Máxima €/h	32 €	64 €	+138 €

Cuadro 3.3: Tarifas de Desarrollador de Python

[2]Ignoramos el IVA ya que se suele cobrar y declarar por separado, viéndose compensado.

	Coste
Tarifa Bruta Media por Hora	25 €/h
Total Bruto	9000 €
Neto por Hora	~18,75 €/h
Total Neto	~6750 €

Cuadro 3.4: Total del Desarrollador de Python

Aplicando el mismo proceso de estimación realizada como analista, podremos estimar el **coste total como desarrollador** *(Cuadro 3.4)*.

3.3.2. Presupuesto Total

Una vez contabilizado lo anterior, estaríamos hablando de un **presupuesto del trabajo** *(Cuadro 3.5)* realizado de:

	Analista	**Desarrollador**	**Total**
Coste	6840 €	9000 €	*15840* €

Cuadro 3.5: Presupuesto Total

Parte II

Documentación técnica

Capítulo 4

Análisis

4.1. Casos de Uso

A continuación se deja detallado los casos de uso contemplados del proyecto DACAI.

Caso de Uso	CU-01. Introducción de los datos hemográficos del paciente
Actor	Usuario
Descripción	El usuario introducirá los resultados del análisis de sangre del animal, se guardarán y se mostrarán en pantalla.
Precondiciones	PRE-1. El usuario deberá tener los resultados del hemograma.
Postcondiciones	POST-1. Los datos son guardados y mostrados.
Flujo normal	FN1 El actor introduce los datos del animal uno por uno. FN2 El sistema comprueba los datos introducidos uno por uno. FN3 Si los datos son correctos, el sistema al final los almacena y lo muestra en pantalla para validación del usuario.
Flujo alternativo	FA3 Si el dato introducido no es un valor válido, mostrará un mensaje de error solicitando volver a introducirlo.
Excepciones	-
Prioridad	Alta
Observaciones	El usuario deberá tener en cuenta que los datos introducidos deben corresponderse con las medidas generales utilizadas en máquinas como ProCyte Dx[23]

Cuadro 4.1: CU-01. Introducción de los datos hemográfico del paciente

Caso de Uso	CU-02. Predicción de la especie
Actor	SpecAI
Descripción	SpecAI, con los datos introducidos, determinará la especie del animal.
Precondiciones	PRE-1. El usuario deberá de haber introducido anteriormente los datos del hemograma del animal en cuestión.
Postcondiciones	POST-1. Se mostrará en pantalla la predicción de SpecAI.
Flujo normal	FN1 SpecAI analiza los datos introducidos. FN2 SpecAI devolverá su predicción entre Canino o Felino.
Flujo alternativo	-
Excepciones	-
Prioridad	Alta
Observaciones	Internamente se descarta aquellos datos innecesarios a entregar a SpecAI para su predicción.

Cuadro 4.2: CU-02. Predicción de la especie

Caso de Uso	CU-03. Predicción del diagnóstico presuntivo
Actor	BlooDAI
Descripción	BlooDAI determinará la especie del animal.
Precondiciones	PRE1 El usuario deberá de haber introducido anteriormente los datos del hemograma del animal en cuestión. PRE2 Anteriormente se debe de haber usado SpecAI para determinar la especie.
Postcondiciones	POST-1. Se mostrará en pantalla la predicción de BlooDAI.
Flujo normal	FN1 BlooDAI analiza los datos. FN2 BlooDAI devolverá su predicción de diagnóstico.
Flujo alternativo	-
Excepciones	-
Prioridad	Alta
Observaciones	Internamente se descarta aquellos datos innecesarios a entregar a BlooDAI para su predicción.

Cuadro 4.3: CU-03. Predicción del diagnóstico presuntivo

4.2. Atributos de calidad

Entre las IAs, podemos destacar:

Portabilidad y Adaptabilidad: Se puede usar en cualquier ordenador con cualquier hardware. Sólo habrá un cambio de desempeño en caso de querer entrenar las IAs desde 0, pero no lo hace imposible.

Rendimiento: La respuesta de las IAs se genera en un intervalo inferior a 1s, y la capacidad total requerida será de apenas 35MB.

Seguridad: En ningún momento se recogerá datos sensibles del paciente y ni se verán almacenados permanentemente.

Fiabilidad: SpecAI se puede probar en situaciones reales sin apenas un ratio de fallo.

Usabilidad: El usuario puede determinar la especie, y el diagnóstico presuntivo en la medida de lo posible.

- **Reconocimiento de idoneidad:** El análisis de sangre es uno de las pruebas más realizadas en las consultas, pudiendo agilizar su trabajo.
- **Capacidad de aprendizaje:** En el momento que se quiera usar, el fichero de ejecución cuenta con guías paso a paso para que el usuario vaya entendiendo lo que está haciendo y no solo de su uso final, sino también de su desarrollo.

Mantenibilidad:

- **Capacidad para ser modificado:** En caso de querer ampliar los casos de entrenamiento para mejorar las IAs, a partir de la recolección de datos en un fichero CSV tipo al facilitado, sólo deberá seguir los pasos indicados en el fichero de entrenamiento.
- **Capacidad para ser probado:** Para exportar las IAs, se tienen que entrenar y probar. En el fichero correspondiente de JupyterNotebook (*<modeloAI>Development* o *DACAI*) se facilita información para probar.

Capítulo 5

Diseño

5.1. Diccionario de Datos

La **información** que se extrae de un **análisis de sangre (hemograma)**[23] *(Cuadro 5.1, Cuadro 5.2 y Cuadro 5.3)* generalmente es la siguiente:

Glóbulos Rojos (Eritrocitos)	
Eritrocitos	Número total de glóbulos rojos por volumen de sangre. Evalúa la capacidad de transporte de oxígeno.
HCT (Hematocrito)	Porcentaje del volumen sanguíneo ocupado por los eritrocitos. Útil para diagnosticar anemia o deshidratación.
HGB (Hemoglobina)	Concentración de hemoglobina en sangre. Indica la capacidad de oxigenación.
MCV (Volumen Corpuscular Medio)	Tamaño promedio de los eritrocitos. Ayuda a clasificar anemias como microcítica o macrocítica.
MCH (Hemoglobina Corpuscular Media)	Cantidad promedio de hemoglobina por eritrocito.
MCHC (Concentración Media de Hemoglobina Corpuscular)	Relación entre hemoglobina y volumen celular. Indica si los eritrocitos están hipocrómicos o normocrómicos.
RDW (Distribución del Volumen Eritrocitario)	Mide la variabilidad en el tamaño de los eritrocitos. Elevado en anemias regenerativas.
%RETIC	Porcentaje de reticulocitos (eritrocitos inmaduros). Refleja la actividad de la médula ósea.
RETIC	Número absoluto de reticulocitos. Indica si la producción de eritrocitos está aumentada.
RET-HE	Contenido de hemoglobina en los reticulocitos. Útil para evaluar el estado del hierro.

Cuadro 5.1: Parámetros de los glóbulos rojos en sangre

Glóbulos Blancos (Leucocitos)	
Leucocitos	Recuento total de glóbulos blancos. Indica presencia de infección, inflamación o enfermedades inmunitarias.
%NEU	Porcentaje de neutrófilos.
%LYM	Porcentaje de linfocitos.
%MONO	Porcentaje de monocitos.
%EOS	Porcentaje de eosinófilos.
%BASO	Porcentaje de basófilos.
NEU	Número absoluto de neutrófilos. Elevado en infecciones bacterianas o estrés.
LYM	Número absoluto de linfocitos. Aumenta en infecciones virales o reacciones inmunes.
MONO	Número absoluto de monocitos. Elevado en inflamación crónica o infecciones persistentes.
EOS	Número absoluto de eosinófilos. Aumenta en alergias o infestaciones parasitarias.
BASO	Número absoluto de basófilos. Rara vez elevados; pueden indicar reacciones alérgicas o enfermedades parasitarias.

Cuadro 5.2: Parámetros de los glóbulos blancos en sangre

Plaquetas	
PLQ (Plaquetas)	Recuento total de plaquetas. Esencial para la coagulación. Valores bajos pueden causar hemorragias.
MPV (Volumen Plaquetario Medio)	Tamaño promedio de las plaquetas. Ayuda a evaluar su producción y destrucción.
PCT (Plaquetocrito)	Porcentaje del volumen sanguíneo ocupado por plaquetas. Análogo al hematocrito para glóbulos rojos.
PDW (Amplitud de Distribución Plaquetaria)	Indica la variabilidad en el tamaño de las plaquetas, ayudando a detectar alteraciones en su producción o destrucción. Estudiado normalmente en caninos.

Cuadro 5.3: Parámetros plaquetarios en sangre

Los **parámetros porcentuales** sirven como ayuda para el estudio de los valores absolutos obtenidos en el resto de parámetros, por lo que solo tienen una **función informativa** a la hora de realizar el diagnóstico presuntivo al paciente (vistos en *Cuadro 5.1 y Cuadro 5.2*).

Dependiendo de los valores absolutos obtenidos, se podrá determinar el estado del paciente y su patología si aplica.

Capítulo 6

Implementación

6.1. Requisitos Técnicos

A continuación se detallan las distintas **librerías** básicas usadas junto con las versiones correspondientes utilizadas *(Cuadro 6.1)*. Es **recomendable** tener las mismas versiones si es posible, pero nunca inferiores.

Librería	Descripción	Versión
numpy	Librería que proporciona estructuras de datos eficientes y funciones para operar con arreglos y matrices multidimensionales.	1.24.3
pandas	Librería para manipulación y análisis de datos mediante estructuras como DataFrame y Series.	2.0.3
seaborn	Librería para visualización estadística basada en Matplotlib, con gráficos atractivos y de alto nivel.	0.12.2
matplotlib	Librería básica de visualización en Python que permite crear gráficos estáticos, animados e interactivos.	3.7.2
scikit-learn	Librería para aprendizaje automático que incluye herramientas para clasificación, regresión, clustering y reducción de dimensionalidad.	1.3.0

Cuadro 6.1: Librerías esenciales

Se decide trabajar con ***scikit-learn***[28] ya que no sólo es bueno para empezar a programar y estructurar los datos con intenciones de análisis y estadística, sino que aporta **algoritmos de última generación**, sin contar con la **mejor documentación** de las bibliotecas de **código abierto** existentes actualmente. Esto la hace calificarse como muy destacable para realizar modelos predictivos en vista de **ponerse en producción**.

A parte, la versión de lenguaje de **Python** con el que se ha desarrollado es el **3.11.5**.

6.2. Análisis y Extracción de Casos Reales

Como se puede observar en el diccionario de datos, contamos ya con un total de **25 parámetros** a analizar de base, que habrá que sumar un parámetro más para *BlooDAI* al tener que recoger el resultado obtenido de *SpecAI*, la **especie**. Esto es un punto preocupante al contar con la gran **limitación en la extracción** de registros reales.

La **gestión de las relaciones con clientes (*CRM*)**[34] que utilizan las clínicas veterinarias no están preparadas para una extracción de los datos de interés ni constan de los filtrados necesarios para ello (no existe ninguna entidad de relación que conecte al paciente y sus resultados de manera directa).

Es por ello que la extracción de los datos ha tenido que ser de manera individual y manual, alcanzando un total de **150 muestras**.

Esta información deberá ser relacionada con los resultados obtenidos en la muestra por la cual se clasificará. Las clasificaciones posibles dependerán de la IA del cual estemos trabajando, por lo tanto:

- **SpecAI: 2 clases** posibles.
  - Canino (88)
  - Felino (62)
- **BlooDAI: 17 clases** posibles.
  - Normal (46)
  - Otros (17)
  - Linfopenia (17)
  - Reticulocitosis (9)
  - Anemia no regenerativa (8)
  - Neutrofilia extrema (7)
  - Trombocitopenia (7)
  - Anemia regenerativa (6)
  - Deshidratación (5)
  - Neutropenia (5)
  - Eritrocitosis (4)
  - Leucograma inflamatorio agudo (4)
  - Trombocitosis (4)
  - Leucograma de estrés (3)

- Linfocitosis (3)
- Neutrofilia (3)
- Leucograma inflamatorio hiper agudo (2)

Estos datos **no se encuentran balanceados** (no hay la misma cantidad de casuísticas en las distintas clases). Aunque depende mucho del algoritmo usado, por lo general en aquellas con mayor cantidad de muestras se obtiene una mejor precisión (*Normal, Linfopenia*) al contrario de con aquellas que se consta de menos (*Linfocitosis, Leucogramas*).

Esto es un punto muy importante a considerar a la hora de plantear los distintos entrenamientos con los algoritmos de aprendizaje que vayamos a probar.

Por intentar mejorar el desempeño, sobre todo en **BlooDAI**, se hizo una **ampliación extra manual de 81 muestras** con los conocimientos adquiridos gracias a la interacción directa de las veterinarias y **revisado** posteriormente por ellos, siendo también **inferior a los caso reales extraídos**.

6.3. Limpieza y Tratamiento de los Datos: Atributos Relevantes

Antes de cualquier comienzo de un entrenamiento, es esencial revisar de nuevas todos esos datos y optimizar los datos con una limpieza adecuada.

Esta **limpieza** siempre es recomendable comenzar detectando primero si se consta de **valores nulos**, y si los hay, descubrir la naturaleza de ello. En este caso, los parámetros con valores nulos son aquellos que tienen un enfoque informativo, comentado anteriormente en el diccionario de datos, y se pueden descartar sin problemas. Esto es fácilmente realizado a partir de un método incluido en la librería de *pandas*:

```
<dataset>.dropna(axis='columns')
```

Gracias a esto, **reducimos a 19 parámetros**. Es una mejoría notable, pero esto se puede mejorar identificando aquellos atributos (*features*) que nos ayuden a hacer la predicción en los respectivos modelos de IA. Esto se puede hacer gracias a la librería *seaborn* mediante la generación de un **gráfico de dispersión**, mostrando las **relaciones** de cada uno de los atributos con ellos mismos a partir de las distintas clasificaciones.

```
<seaborn>.pairplot(<dataset>, hue="<clave_clasificatoria>",
diag_kind="kde", markers=["o", "s"])
```

A continuación veremos los gráficos de dispersión generados en cada desarrollo, empezando por *SpecAI*.

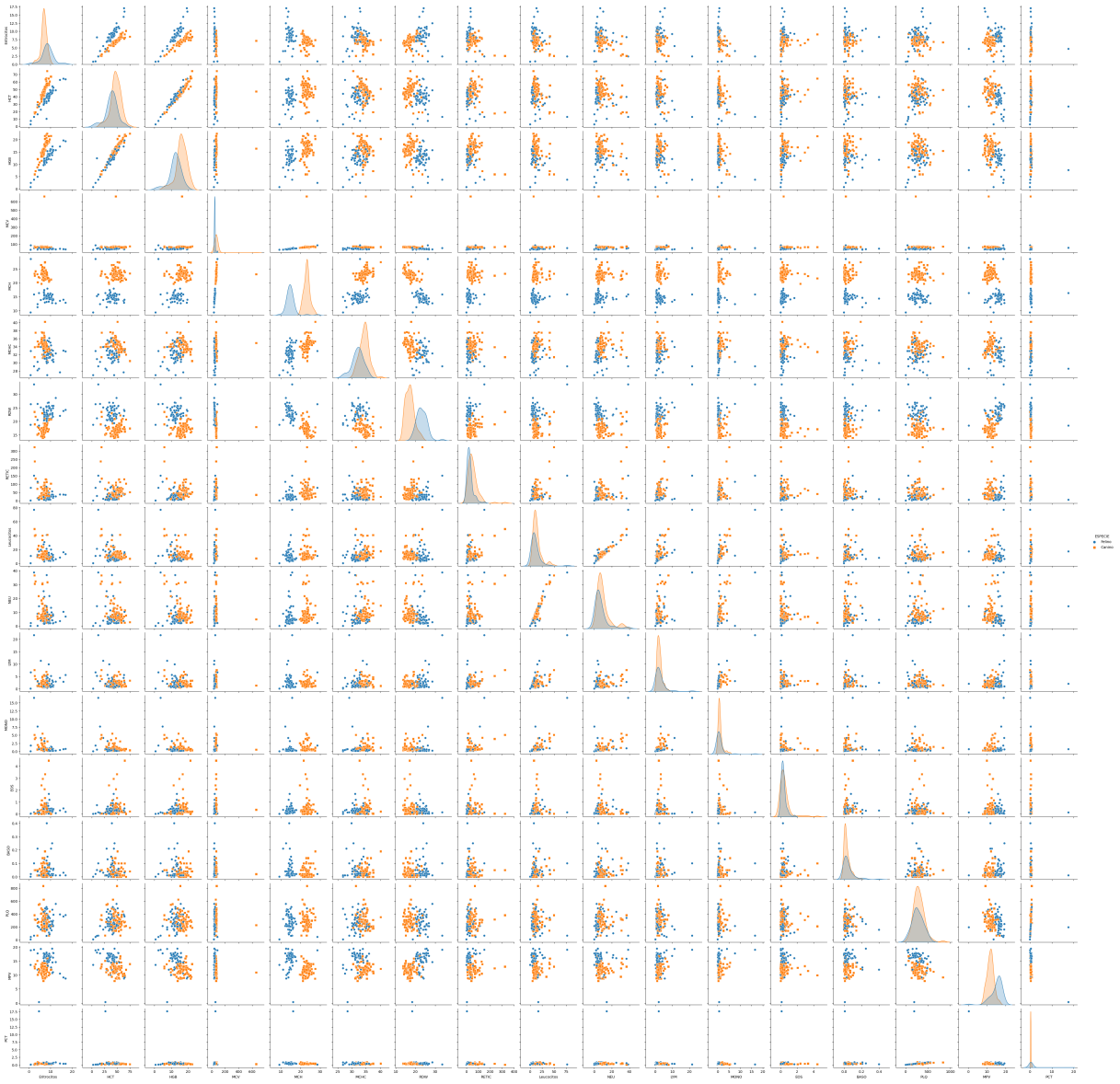

Figura 6.1: Gráfico de dispersión por Especie

En el caso de determinar la especie nos encontramos con un **caso ideal**. Esto es porque con el gráfico de dispersión (*Figura 6.1*) podemos ver aquellos que son capaces de diferir o **separar claramente entre ambas especies**, cosa que normalmente no es así como veremos a continuación en el caso del diagnóstico, *BlooDAI*.

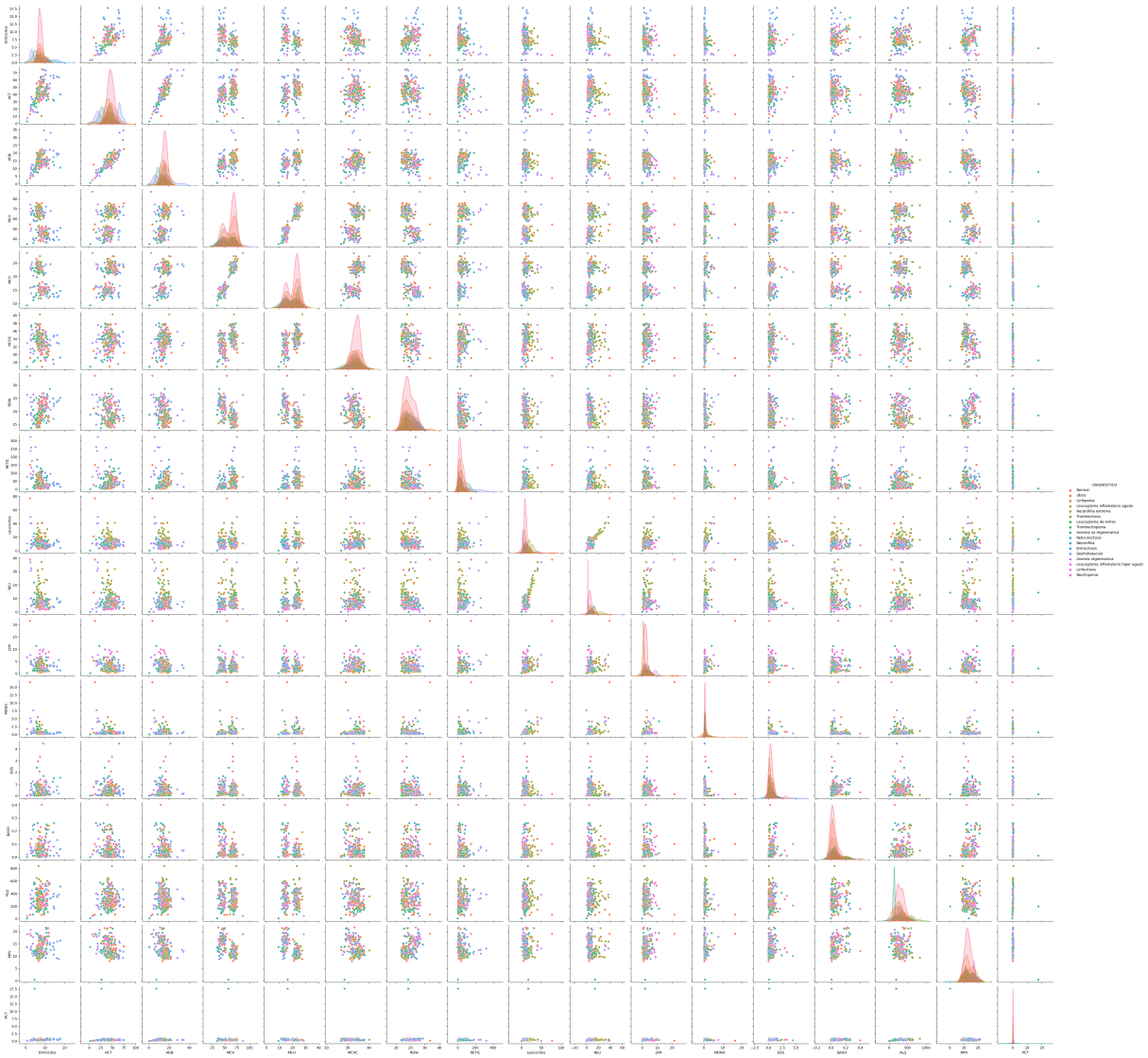

Figura 6.2: Gráfico de dispersión por Diagnóstico Presuntivo

Para diagnosticar *(Figura 6.2)* al paciente podemos ver que todos las características o atributos están **intrínsecamente relacionados**, impidiendo detectar los más relevantes y con los que centrarnos a preparar y entrenar el modelo. Esto es lo más común a encontrarse con un conjunto de datos (*dataset*) y más con una gran cantidad de clasificaciones.

En estos casos debemos utilizar alguna técnica para poder determinar las características relevantes mediante la **definición de un criterio** que tienen que cumplir.

6.3.1. Matriz de Correlación

Tenemos la **ventaja de que nuestros datos son valores numéricos** (a excepción de las clases, que requerirán transformarlas), que esto nos permitirá utilizar más fácilmente una técnica matemática en la cual podamos **visualizar la relevancia o la importancia de los atributos** con respecto a la clase clasificatoria, ya sea para la especie o para el diagnóstico. Estamos hablando de la matriz de correlación.

La **matriz de correlación** es una tabla en la que muestra los coeficientes de correlación entre todos los parámetros y la clase para **determinar su dependencia y relevancia** en el dataset. Esto luego nos permitirá determinar un criterio mínimo de relación que deberán tener nuestros parámetros y descartar aquellos que no los cumplan.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{6.1}$$

Ecuación 6.1: Coeficiente de correlación de Pearson.

Una vez transformadas las clases en un valor numérico, utilizaremos el método ***corr*** sobre el dataset para obtener esa representación matricial (*Figura 6.3*).

```
cor = <dataset-num>.corr()
```

	Eritrocitos	HCT	HGB	MCV	MCH	MCHC	RDW	RETIC	Leucocitos	NEU	LYM	MONO	EOS	BASO	PLQ	MPV	PCT	ESPECIE
Eritrocitos	1.000000	0.639894	0.556923	-0.114643	-0.416074	-0.168414	0.353223	-0.303323	-0.341273	-0.303181	-0.158686	-0.398934	0.059080	0.016697	0.074658	0.069329	-0.093150	0.353826
HCT	0.639894	1.000000	0.947076	0.096978	0.298138	0.096799	-0.257422	-0.078611	-0.322714	-0.255965	-0.268795	-0.382287	0.161028	-0.069148	0.142927	-0.357774	-0.119780	-0.360555
HGB	0.556923	0.947076	1.000000	0.133879	0.436253	0.296978	-0.375900	-0.081619	-0.293994	-0.224967	-0.255793	-0.376492	0.197878	-0.087069	0.082631	-0.357769	-0.159797	-0.473155
MCV	-0.114643	0.096978	0.133879	1.000000	0.284099	0.145832	-0.209315	0.061655	-0.018433	-0.004393	-0.046827	-0.009937	0.013928	-0.065391	-0.015521	-0.178766	-0.022082	-0.272329
MCH	-0.416074	0.298138	0.436253	0.284099	1.000000	0.602152	-0.772118	0.306634	0.117113	0.157172	-0.033447	0.044132	0.111411	-0.143077	0.081864	-0.472700	-0.090197	-0.897610
MCHC	-0.168414	0.096799	0.296978	0.145832	0.602152	1.000000	-0.467227	0.028158	0.061218	0.081907	-0.020360	-0.014937	0.083845	-0.098603	-0.024011	-0.146651	-0.204498	-0.519450
RDW	0.353223	-0.257422	-0.375900	-0.209315	-0.772118	-0.467227	1.000000	-0.001357	0.047639	-0.055989	0.283053	0.188153	-0.178399	0.149821	-0.153453	0.583976	-0.016798	0.780110
RETIC	-0.303323	-0.078611	-0.081619	0.061655	0.306634	0.028158	-0.001357	1.000000	0.417982	0.336773	0.345537	0.417674	0.017912	-0.045967	0.023883	-0.072976	-0.094342	-0.303956
Leucocitos	-0.341273	-0.322714	-0.293994	-0.018433	0.117113	0.061218	0.047639	0.417982	1.000000	0.937585	0.645729	0.769291	-0.052024	0.138417	-0.014559	-0.000013	0.026247	-0.132344
NEU	-0.303181	-0.255965	-0.224967	-0.004393	0.157172	0.081907	-0.055989	0.336773	0.937585	1.000000	0.380250	0.561717	-0.120364	0.113037	0.053416	-0.080621	0.054272	-0.177248
LYM	-0.158686	-0.268795	-0.255793	-0.046827	-0.033447	-0.020360	0.283053	0.345537	0.645729	0.380250	1.000000	0.731367	-0.008243	0.099789	-0.147685	0.208922	-0.035084	0.070126
MONO	-0.398934	-0.382287	-0.376492	-0.009937	0.044132	-0.014937	0.188153	0.417674	0.769291	0.561717	0.731367	1.000000	-0.041484	0.109585	-0.143327	0.154154	-0.035010	-0.051943
EOS	0.059080	0.161028	0.197878	0.013928	0.111411	0.083845	-0.178399	0.017912	-0.052024	-0.120364	-0.008243	-0.041484	1.000000	0.060592	-0.031838	-0.058855	-0.046025	-0.129956
BASO	0.016697	-0.069148	-0.087069	-0.065391	-0.143077	-0.098603	0.149821	-0.045967	0.138417	0.113037	0.099789	0.109585	0.060592	1.000000	-0.037495	0.079637	-0.040064	0.143498
PLQ	0.074658	0.142927	0.082631	-0.015521	0.081864	-0.024011	-0.153453	0.023883	-0.014559	0.053416	-0.147685	-0.143327	-0.031838	-0.037495	1.000000	-0.318350	0.030485	-0.105973
MPV	0.069329	-0.357774	-0.357769	-0.178766	-0.472700	-0.146651	0.583976	-0.072976	-0.000013	-0.080621	0.208922	0.154154	-0.058855	0.079637	-0.318350	1.000000	-0.350986	0.556606
PCT	-0.093150	-0.119780	-0.159797	-0.022082	-0.090197	-0.204498	-0.016798	-0.094342	0.026247	0.054272	-0.035084	-0.035010	-0.046025	-0.040064	0.030485	-0.350986	1.000000	0.122633
ESPECIE	0.353826	-0.360555	-0.473155	-0.272329	-0.897610	-0.519450	0.780110	-0.303956	-0.132344	-0.177248	0.070126	-0.051943	-0.129956	0.143498	-0.105973	0.556606	0.122633	1.000000

Figura 6.3: Matriz de Correlación durante el desarrollo de SpecAI

La correlación **mide la fuerza y dirección de la relación lineal entre dos variables**, con valores que van de -1 (correlación negativa perfecta) a 1 (correlación positiva

perfecta). Un valor de 0 indica que no hay relación lineal. Por esta razón, es **recomendable convertirlos en valores absolutos**, que esto luego nos permitirá **definir un criterio** por un valor mínimo de relevancia, obteniendo así las características más relevantes para nuestro modelo.

```
cor_Target =  abs(cor["ESPECIE"])
relevant_features = cor_Target[cor_Target>0.5]
relevant_features
-----
MCH         0.897610
MCHC        0.519450
RDW         0.780110
MPV         0.556606
ESPECIE     1.000000
```

Este punto es fuertemente beneficiado *SpecAI*, reduciendo a tan **sólo 4 atributos**[1] de los 26 (25 iniciales + *Id*) que contábamos. Esto fácilmente lo podemos verificar con la visualización de la **distribución de las variables** mediante la librería *seaborn* (*Figura 6.4*).

```
ax = sns.boxplot(x="ESPECIE", y="MCH", data=<dataset>)
ax = sns.stripplot(x="ESPECIE", y="MCH", data=<dataset>,
jitter=0.25, edgecolor="gray")
```

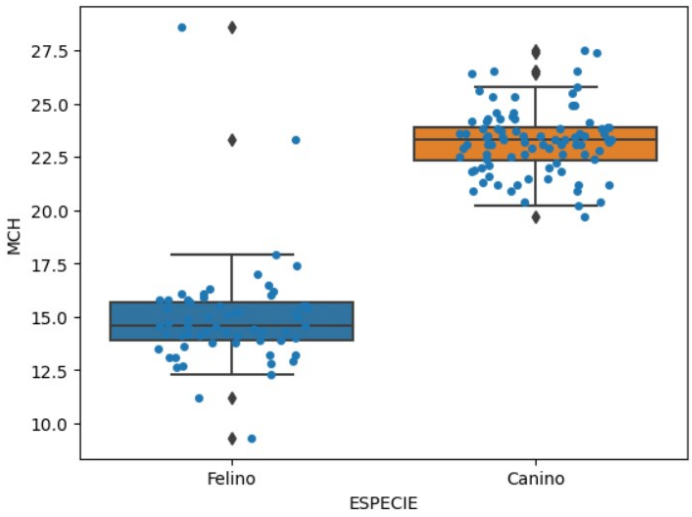

Figura 6.4: Distribución de *MCH* en el desarrollo de SpecAI

[1]Hay que descartar la clase como característica relevante ya que es lo que queremos predecir, y por eso tiene una relación del 100 %

El ***boxplot*** proporciona una visión general de la dispersión y los valores atípicos, mientras que el ***stripplot*** da un detalle más fino, permitiendo ver la densidad de puntos dentro de cada clase. Si las franjas entre ellas no se solapan o muy poco, se demostrará que son características adecuadas para predecir su resultado.

Sin embargo, *BlooDAI* se quedará corto implementando esta técnica a raíz del gran volumen de casuísticas a considerar, bajando a **16 atributos**.

```
cor_Target =  abs(cor["DIAGNOSTICO"])
relevant_features = cor_Target[cor_Target>0.07]
relevant_features
-----
Eritrocitos    0.128443
HCT            0.097668
HGB            0.139860
MCV            0.247962
MCH            0.236359
MCHC           0.172870
RDW            0.247340
RETIC          0.214440
Leucocitos     0.084714
NEU            0.126728
LYM            0.273706
EOS            0.080732
BASO           0.146129
PLQ            0.077293
MPV            0.133751
ESPECIE        0.213824
DIAGNOSTICO    1.000000
```

Esto es debido a que gracias al conocimiento adquirido sabemos que dos diagnósticos (*Trombocitosis y Trombocitopenia*) están fuertemente ligados por el valor de las plaquetas (***PLQ***), por lo que como mínimo necesitamos un criterio en el que se vea contemplado.

En el atributo más relevante se podrá ver la fuerte dependencia que hay en general para determinar el diagnóstico presuntivo (*Figura 6.5*).

Ya con las características relevantes detectadas, podemos proceder a la preparación del entrenamiento con los distintos algoritmos de aprendizaje, no sin antes actualizar el dataset descartando las características que no cumplen el criterio definido.

```
X = <dataset>.loc[:, [<relevant_features>]]
Y = <dataset>[<dataset>.columns[-1]] # La clase coincide con la última
posición del dataset
```

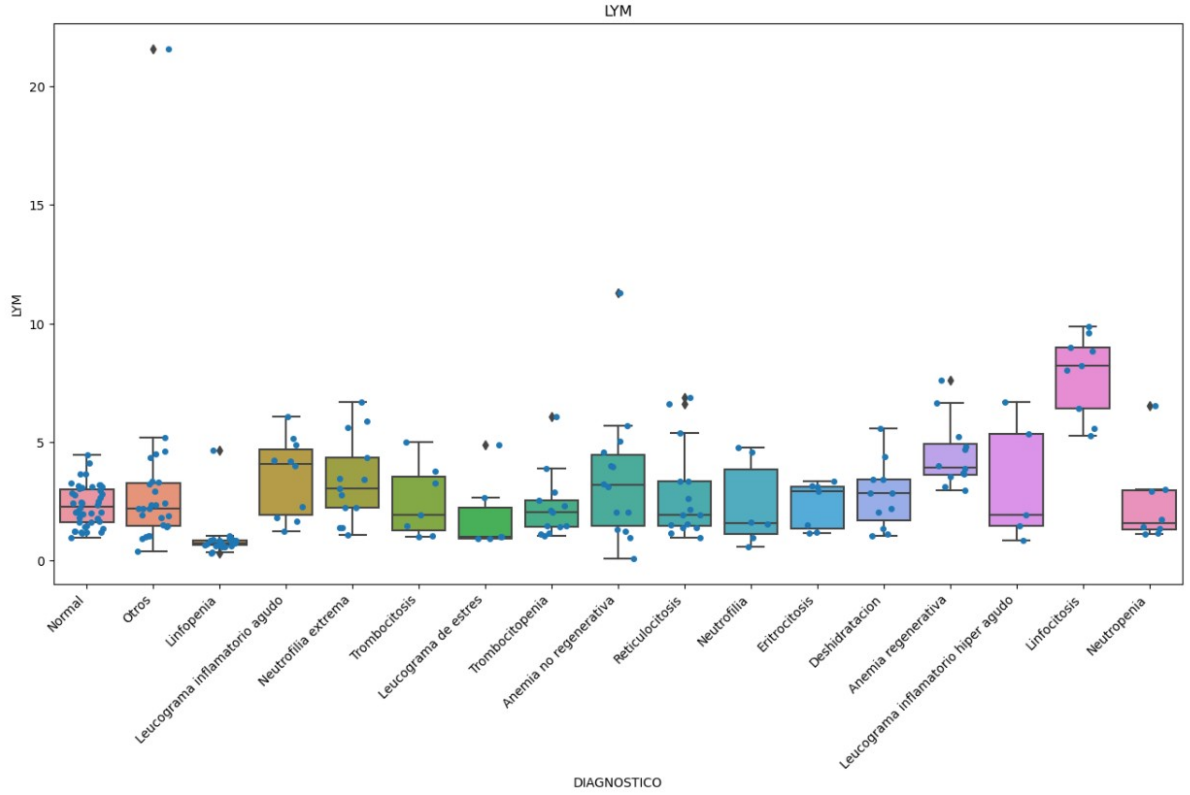

Figura 6.5: Distribución de *LYM* en el desarrollo de BlooDAI

De esta manera, hacemos 2 cosas:

- Definimos el **conjunto de datos** con el que la IA tendrá que predecir o clasificar ($\boldsymbol{X}$).
- Y los **valores de la predicción** o clasificación que deberá predecir la IA con respecto al conjunto de datos ($\boldsymbol{Y}$).

6.4. Entrenamiento de las IAs

Una vez que hemos optimizado lo máximo posible estos datos, empezaremos a distribuirlos en **conjuntos de entrenamiento (*train*) y pruebas (*testing*)** para hacer una primera toma de contacto con el desempeño entre los distintos algoritmos de aprendizaje supervisado[1] mediante la librería ***sklearn***.

```
trainX, testX, trainY, testY = train_test_split(X, Y, random_state=42,
test_size=0.3, stratify=Y)
```

Lo que hacemos con esto es utilizar la función ***train_ test_ split***, que nos permite **dividir un conjunto de datos en dos subconjuntos**: uno para entrenar el modelo (train) y otro para evaluarlo (test). Para hacer esa separación hay que especificarlo mediante los **hiperparámetros**:

- **X:** Conjunto de características (variables independientes).
- **Y:** Conjunto de etiquetas (variable dependiente o la clase que queremos predecir).
- ***random_ state*=42:**[2] Establece la **semilla** para la generación de números aleatorios, lo que asegura que la división **sea replicable** y poder hacer un **análisis de desempeño** entre distintos modelos.
- ***test_ size*=0.3:** En este caso, especificamos que el tamaño del conjunto de prueba es el 30 % del total de los datos, y el resto (70 %) será usado para el entrenamiento.
- ***stratify*=Y:** La **estratificación** asegura que las clases en el conjunto de entrenamiento y prueba tengan la **misma distribución que en el conjunto original**. Esto es especialmente importante ya que, como hemos visto anteriormente, contamos de un número **desbalanceado** de casos, permitiendo **evitar que el modelo se entrene con una distribución sesgada** de las clases.

Una vez realizado esto, podemos empezar a hacer pruebas con varios algoritmos de aprendizaje supervisado y encontrar el mejor para cada uno de nuestros modelos objetivo, *SpecAI* y *BlooDAI*.

Antes de ver los algoritmos usados, primero entendamos cómo podemos analizar y comparar entre ellos mediante sus resultados durante sus pruebas.

6.4.1. Proceso de Entrenamiento y Comprensión de los Resultados en las Pruebas

En cada algoritmo o modelo que se quiera usar, primero deberá ser entrenado con el conjunto de datos de entrenamiento para después predecir con el conjunto de pruebas (previamente definidos).

Aunque es necesario saber la precisión global del modelo en ambos casos, es erróneo solamente analizarlo por ello. El método ***classification_ report*** nos otorga una información profunda a partir del **rendimiento** obtenido mediante las siguientes métricas:

[2]Este valor puede ser cualquiera, pero es muy reconocido en Machine Learning gracias a la popularización del libro "The Hitchhiker's Guide to the Galaxy"de Douglas Adams en 1978

- ***recall* (Sensibilidad o Exhaustividad):** Mide la capacidad del modelo para identificar correctamente todas las **instancias positivas**. Es el porcentaje de verdaderos positivos (TP) sobre el total de instancias que realmente son positivas (verdaderos positivos + falsos negativos).

  Fórmula: $\frac{TP}{TP+FN}$

- ***F1-score*:** Es la **media** armónica entre la **precisión** (*precision*) y el ***recall***. **Se utiliza cuando hay un desequilibrio entre las clases**, ya que combina ambos aspectos en una sola métrica, algo muy importante sobre todo para el caso de *BlooDAI*.

  Fórmula: $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

- ***macro avg*: Promedia las métricas de rendimiento** (*precision*, *recall*, *F1-score*) de todas las clases, tratando cada clase por igual, independientemente de su frecuencia. Esto hace que en **clases desbalanceadas se les de el mismo peso o importancia**, permitiendo una **evaluación más justa del rendimiento** en todas las clases evitando sesgos.

Entendámoslo con un **ejemplo sencillo** realizado en el proyecto, en concreto para el desarrollo de *SpecAI*.

```
# Entrenamos el modelo
modelKNN_1= KNeighborsClassifier(n_neighbors=1)
modelKNN_1.fit(trainX, trainY)

print("Precisión durante el ENTRENAMIENTO (TRAIN Accuracy):
{0:.2%}".format(modelKNN_1.score(trainX, trainY)))
print("Precisión durante las PRUEBAS (TEST Accuracy):
{0:.2%}".format(modelKNN_1.score(testX, testY)))

# Hacemos predicciones con los datos de test
predictionsKNN_1 = modelKNN_1.predict(testX)

print(classification_report(testY, predictionsKNN_1,
target_names=dataset_class_names))
```

```
----
```

```
Precisión durante el ENTRENAMIENTO (TRAIN Accuracy): 100.00%
Precisión durante las PRUEBAS (TEST Accuracy): 97.78%
```

	precision	recall	f1-score	support
Felino	0.96	1.00	0.98	26
Canino	1.00	0.95	0.97	19
accuracy			0.98	45
macro avg	0.98	0.97	0.98	45
weighted avg	0.98	0.98	0.98	45

En este caso, podemos ver de manera general que la precisión por ambas partes, entrenamiento y pruebas, ha sido bastante positiva. Sin embargo, con una evaluación más profunda podemos ver un desempeño más detallado para determinar finalmente si ha sido tan exitoso como nos lo quiere "pintar".

En el caso del **Felino**:

- ***precision*:** De todas las predicciones en las que el modelo predijo que la clase era Felino, el **96 %** de ellas fueron correctas.
- ***recall*:** De todos los casos reales de la clase Felino, el modelo predijo correctamente **todos** los casos. **No hay falsos negativos**. En este caso no hay nada por lo que preocuparse. Veremos en el caso del Canino.
- ***f1-score*:** Combinación de *precision* y *recall*. Un F1-score de **0.98** es excelente y muestra un **buen balance**.

En el caso del **Canino**:

- ***precision*:** De todas las predicciones en las que el modelo predijo que la clase era Canino, **todas fueron correctas**.
- ***recall*:** El modelo predijo correctamente el **95 %** de los casos reales de la clase Canino, pero tuvo **algunos falsos negativos**. Esto quiere decir que clasificó como Felino en vez de Canino. Si lo ponemos en ejemplo de un estudio para detectar si tienes cáncer, significa que puede haber decidido que no tienes cáncer cuando lo tienes. Es muy importante tener esto en consideración al escoger el mejor modelo y más aún en el ámbito de la salud.
- ***f1-score*:** Similar al caso anterior, el F1-score de **0.97** indica un **buen rendimiento** en términos de *precision* y *recall* combinados.

Si observamos los **promedios**:

- ***macro avg (0.98 precision, 0.97 recall, 0.98 f1-score)*:** Estos promedios se calculan sin tener en cuenta el soporte (número de muestras de cada clase). El modelo tiene un rendimiento muy alto tanto en precisión como en recall, lo que indica que está funcionando de manera **equilibrada en ambas clases**.

- ***weighted avg (0.98 precision, 0.97 recall, 0.98 f1-score)*:** Estos promedios ponderados consideran el número de ejemplos de cada clase. Dado que hay 26 ejemplos de Felino y 19 de Canino, el modelo tiene un **rendimiento muy equilibrado** en ambas clases. La puntuación es ligeramente mejor para la clase Felino, ya que tiene más ejemplos.

Podemos ver que es un buen modelo para predecir la especie del animal a partir de los datos que contamos, pero eso no quiere decir que no exista otro algoritmo mejor. Por esta razón, en este proyecto **se probará con una gran variedad de algoritmos** y además **se sacará el mayor partido posible de cada uno**.

Si queremos una **presentación más visual del reporte**, se puede utilizar el método ***confusion_matrix*** apoyado de las librerías *seaborn* y *pandas (Figura 6.6)* de la siguiente manera:

```
# Versión usando Seaborn
from pandas import DataFrame
confm = confusion_matrix(testY, predictionsKNN_1)
plt.figure()
df_cm = DataFrame(confm, index=dataset_class_names,
columns=dataset_class_names)
ax = sns.heatmap(df_cm, cmap='Oranges', annot=True)
```

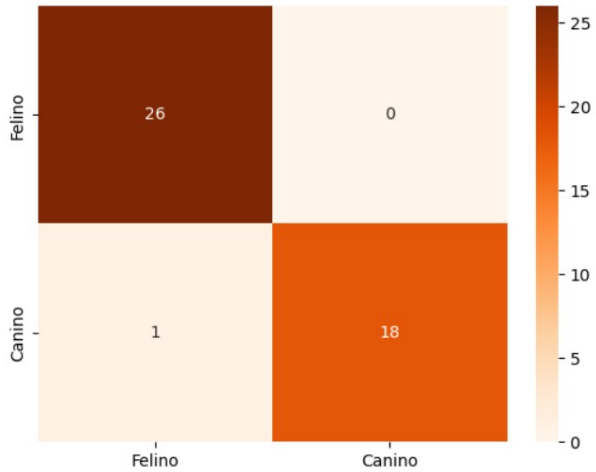

Figura 6.6: Caso Ejemplo de Matriz de Confusión de los resultados de un algoritmo (SpecAI)

Ya entendiendo lo que debemos buscar y analizar de los resultados que vamos a obtener, identifiquemos los algoritmos usados en el proyecto.

6.4.2. Algoritmos de Aprendizaje Supervisado

Los algoritmos utilizados son los siguientes:

1. **K-Vecinos** (*KNN*)[14]
2. **Naive Bayes** - Gausiana[13]
3. **Árboles de Decisión** - Gini y Entropía[9]
4. **Random Forest**[17]
5. **Regresión Logística**[15]
6. **Máquina de Vectores de Soporte** (*SVM*) - Linear y Radial (RBF)[19]
7. **Perceptrón Multicapa** (*MLP*)[16]
8. **Ensembles por Votación**[11]
9. **Aumento de Gradiente** (Gradient Boosting)[10]: Solo en BlooDAI.

K-Vecinos (*K-Nearest Neighboors*)

El primer algoritmo de aprendizaje con el que vamos a empezar es el **K-Vecinos más cercanos** (*K-Nearest Neighbors* - ***KNN***), un algoritmo de aprendizaje supervisado que usa la **clasificación** y la **regresión** basado en la idea de que los **elementos similares están cerca unos de otros** en el espacio de características. Esto lo hace **susceptible al ruido**.

Para clasificar o predecir el valor de un punto desconocido, el algoritmo busca los ***K* puntos más cercanos en el conjunto de datos** de entrenamiento (usando una métrica de distancia como la Euclidiana) y le asigna los valores correspondientes, por lo que:

- **Clasificación:** El punto se asigna a la **clase que es más frecuente** entre sus K vecinos.
- **Regresión:** El **valor predicho** es el **promedio** de los valores de los K vecinos.

Es un buen punto de partida ya que es **sencillo** de entender, pero puede llegar a ser **computacionalmente costoso según** el volumen de los datos.

Debemos **determinar** entonces el mejor valor de ***K***, de lo contrario:

- Si **K es muy pequeño**, el modelo puede volverse muy sensible al ruido (esto puede llevar a **sobreajuste**)
- Si **K es muy grande**, el modelo puede ser demasiado general y perder detalles importantes (lo que puede llevar a un **subajuste**).

Para ello, se hace una **exploración previa mediante la aplicación de un rango definido** de valores y representarlo luego gráficamente los resultados de precisión con el mismo modelo *(Figura 6.7)*.

```
# ¿Y cómo sabemos el número de vecinos a usar?
# Experimentamos con distintos valores de vecinos (entre 1 a 50, por
ejemplo)
k_range = list(range(1,50))
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(trainX, trainY)
    y_pred = knn.predict(testX)
    scores.append(accuracy_score(testY, y_pred))

plt.plot(k_range, scores)
plt.xlabel('Valor de K para KNN')
plt.ylabel('Precisión (accuracy) en la clasificación')
plt.title('Valores de precisión (accuracy) para distintos valores de K
para el algoritmo KNN')
plt.show()
```

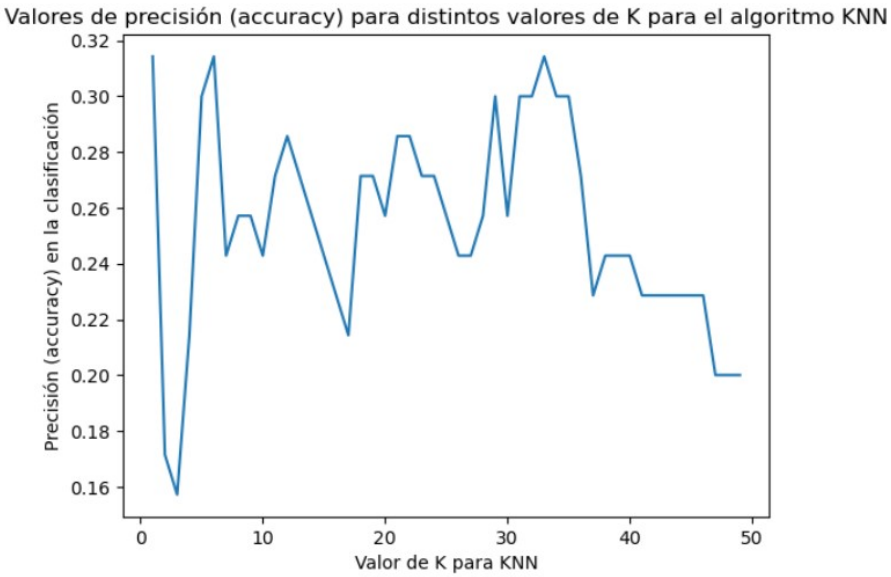

Figura 6.7: Caso Ejemplo de rango de valores de *K* para KNN (BlooDAI)

En este caso intuimos 2 valores que nos ofrecen mejores resultados. Siempre en estas situaciones se recomienda el **valor más pequeño de entre aquellos que lleguen a la máxima precisión** alcanzada.

Naive Bayes

Se trata de un **algoritmo de clasificación probabilístico** basado, como en su nombre indica, en el **Teorema de Bayes**, además de en la suposición de que las características son independientes dada la clase. No es algo común que ocurra a la hora de la verdad, pero sigue siendo bastante buena en casos de clasificación y más si sus características no son complejas.

El *Teorema de Bayes* establece que la probabilidad posterior de una clase C, dado un conjunto de características X, se puede calcular de la siguiente forma:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)} \tag{6.2}$$

Ecuación 6.2: Teorema de Bayes.

Donde:

- $P(C|X)$ es la **probabilidad posterior** de la clase C dado un conjunto de características X.
- $P(X|C)$ es la **probabilidad de observar las características** X, dado que la clase es C.
- $P(C)$ es la **probabilidad a priori** de la clase C, es decir, cuán probable es la clase antes de ver las características.
- $P(X)$ es la **probabilidad total** de las características X, que actúa como una constante en el proceso de clasificación.

Una suposición importante en algunos modelos es que las características son independientes entre sí. Bajo esta suposición, la probabilidad de observar las características X, dado que la clase es C, se puede **descomponer como el producto de las probabilidades individuales** de cada característica:

$$P(X|C) = P(x_1|C) \cdot P(x_2|C) \cdot \cdots \cdot P(x_n|C)$$

Esta suposición de independencia implica que, **para cada característica** x_i, **la probabilidad** de que ocurra dado que la clase es C **es independiente de las demás características**. En términos prácticos, esto simplifica enormemente los cálculos.

Existen **distintos tipos de Naive Bayes** a partir del dato con el que se trabaje:

- **Gaussian Naive Bayes:** Se usa cuando las características son continuas y se asume que siguen una distribución normal (gaussiana).

- **Multinomial Naive Bayes:** Se usa cuando las características son discretas (por ejemplo, el conteo de palabras en clasificación de texto) y se asume que los datos siguen una distribución multinomial.

- **Bernoulli Naive Bayes:** Se usa cuando las características son binarias (por ejemplo, presencia/ausencia de una característica) y se asume que los datos siguen una distribución bernoulliana.

Para ambos modelos de IA que buscamos, se utilizará **Gaussian Naive Bayes**.

Árboles de Decisión

Al igual que K-Vecinos, los **Árboles de Decisión** son un modelo de aprendizaje supervisado utilizado para **tareas de clasificación y regresión**. Su funcionamiento, sin embargo, consiste en **dividir en subgrupos los datos según sus atributos** a partir de ir haciendo "preguntas", de tal forma que progresivamente se vayan **creando nodos** y, por consecuente, **generando hojas**. No se ven limitados por el formato numérico, ya que pueden ser usados también para la categorización.

Tengamos en claro los conceptos básicos de este algoritmo que simularán, a como su nombre da a entender, a un árbol:

- **Raíz:** Nodo inicial donde se divide el conjunto de datos.

- **Nodos Intermedios:** Realizan divisiones sucesivas basadas en características.

- **Hojas:** Contienen la predicción final (clase o valor).

- **Medida de calidad:** Se utiliza una medida como el **índice de Gini** o la **entropía** para elegir la mejor división de los nodos.

Una vez entrenado, se puede **imprimir el árbol** resultante. De esta manera, se obtiene una ganancia en la **interpretación** del modelo. Hay que tener en cuenta, sin embargo, de que como sean **demasiado profundos puede causar mayor complejidad**. En estos casos, se pueden "podar" estos árboles al especificar un máximo de hojas o de nodos intermediarios.

Las medidas de calidad, como bien hemos indicado antes, son:

- **Índice de *Gini*:** Es una medida de impureza utilizada para **evaluar la calidad de una división** en un árbol de decisión *(Figura 6.8)*. Su valor varía **entre 0** (cuando todos los elementos de una división pertenecen a la misma clase) **y 1** (cuando los elementos están igualmente distribuidos entre todas las clases).

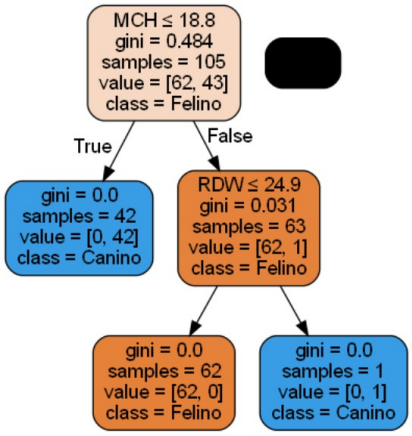

Figura 6.8: Caso Ejemplo de árbol de decisión usando Gini (SpecAI)

- **Entropía:** Es una medida que proviene de la teoría de la información *(Figura 6.9)*. Indica el grado de desorden o impureza de un conjunto de datos, analizando su **aporte de valor clasificatorio o ganancia de información** (la cual hace también ser conocida por ese nombre).

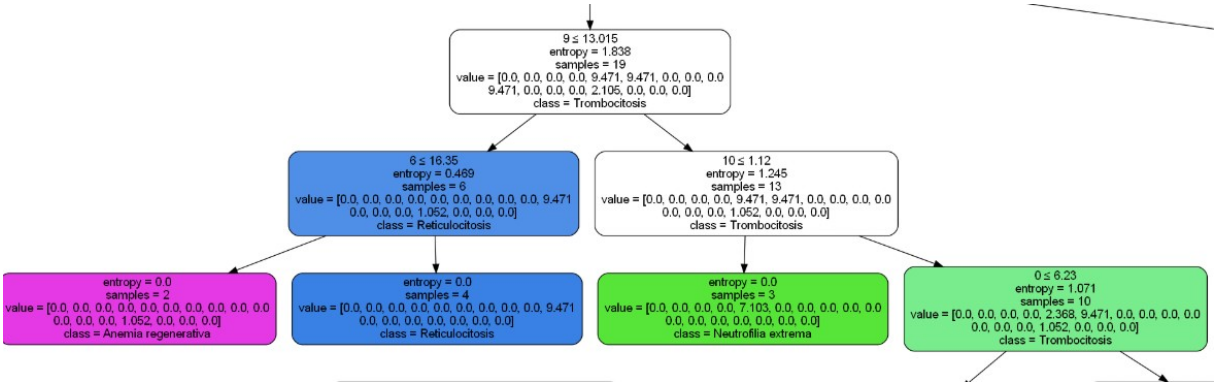

Figura 6.9: Caso Ejemplo de una parte del árbol de decisión usando Entropía (BlooDAI)

Este tipo de algoritmos generará otros nuevos, como el que viene a continuación.

Random Forest

A partir de los Árboles de Decisión existe otro modelo de aprendizaje en el que **combina varios árboles de decisión** para mejorar la **precisión** y la **robustez** además de permitir **calcular la importancia de las características**, conocido como **Random Forest**. Es un algoritmo preparado para conjuntos de datos con una gran variedad de clasificaciones posibles, ideal para *BlooDAI*.

El **procedimiento** que sigue es:

1. **Creación de múltiples árboles:** Random Forest construye varios árboles de decisión de forma independiente. Cada árbol se entrena utilizando una **muestra aleatoria** del conjunto de datos (mediante *Bootstrap sampling*).

2. **Selección aleatoria de características:** En cada nodo de cada árbol, se selecciona de manera aleatoria un subconjunto de características para realizar la división. Esto introduce más **diversidad** entre los árboles y **reduce el riesgo de sobreajuste**.

3. **Predicción:** La predicción final se obtiene mediante **votación** en clasificación (mayoría de votos) **o promedio** en regresión.

Sin embargo, **se pierde la interpretabilidad** del modelo al ser más complejo.

Regresión Logística

Se trata de un modelo que se utiliza principalmente para problemas de clasificación binaria (por ejemplo, predecir "sí" o "no", "positivo" o "negativo"). En realidad, lo que hace es predecir probabilidades.

Usa una **función sigmoide** para transformar una combinación lineal de las características en un valor **entre 0 y 1**. Este valor **representa la probabilidad** de que una muestra **pertenezca a la clase positiva** (por ejemplo, 1). Luego se aplica un **umbral** (generalmente 0.5) **para decidir** la clase.

$$P(y = 1|X) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_n x_n)}} \tag{6.3}$$

Ecuación 6.3: Regresión Logística.

Este modelo es adecuado cuando el **nivel de complejidad es bajo o con pocas características** mientras que las relaciones sean lineales (si no, habría que hacer una transformación), como en el caso de *SpecAI*. Sin embargo, puede **sufrir ante datos desbalanceados**, como en nuestro caso.

Para ajustarla con la mejor **optimización** posible, se minimiza una **función de pérdida (entropía cruzada)** mediante algoritmos de optimización.

- **Gradient Descent (Descenso del Gradiente):** Ajusta los parámetros en la dirección opuesta al gradiente, siendo simple e interpretable para datasets pequeños.
- **Stochastic Gradient Descent (SGD):** Al contrario que el anterior, es rápido para datasets grandes al usar iteraciones a partir de una muestra. Sin embargo, requiere de muchas iteraciones.
- **Mini-Batch Gradient Descent:** Una combinación entre las 2 anteriores, buscando el compromiso entre precisión y eficiencia.

- **Newton-Raphson:** Utiliza la segunda derivada, conocida como Hessiana para la optimización. A cambio de su coste, obtiene una gran precisión.
- **Quasi-Newton (BFGS, L-BFGS):** Además de lo anterior, lo aproxima para ganar eficiencia en memoria. Puede llegar a no escalar bien ante datos dispersos.
- **Adam, RMSProp, Adagrad:** Utilizado para redes neuronales o *Deep Learnning*.

En librerías como *scikit-learn*, como es nuestro caso, esta optimización **se realiza mediante *solvers*** *(Cuadro 6.2)*, que implementan distintas estrategias numéricas.

Solver	**Método Base**	**Multiclase**	**Velocidad**	**Dataset Ideal**	**Observaciones**
newton-cg	Newton-Raphson	Sí	Media	Pequeño / Preciso	Usa Hessiana
lbfgs	Quasi-Newton (L-BFGS)	Sí	Alta	Mediano / Grande	Por defecto en scikit-learn
liblinear	Coordinate Descent	OvR	Alta	Pequeño / Binario	No soporta multinomial
sag	Stochastic Average Grad.	Sí	Muy alta	Grande / Disperso	Necesita datos escalados
saga	SGD mejorado	Sí	Muy alta	Grande / Disperso	Único con soporte para elastic-net

Cuadro 6.2: Solvers en Regresión Logística (*scikit-learn*)

Nos referimos con **Multiclase OvR** a que utiliza una **estrategia para extender en varios modelos binarios** en caso de **un problema con múltiples clases**.

Es decir supongamos, a partir de ***SpecAI*** al ser más simple, que a parte de *Felino y Canino* contamos de más, siendo así: *Canino, Felino, Conejo, Pez*. **Con OvR**, entrenamos 4 modelos binarios:

- **¿Canino vs no canino?**
- **¿Felino vs no felino?**
- **¿Conejo vs no conejo?**
- **¿Pez vs no pez?**

Durante la predicción, **se elige la clase con la mayor probabilidad** entre los 4 modelos.

Support Vector Machines (*SVM*)

Es un algoritmo de aprendizaje supervisado que se utiliza para **clasificación y regresión**. Su objetivo principal es **encontrar un hiperplano óptimo que separe** las clases en el espacio de características **con el mayor margen posible**. Eso es definido mediante vectores de soporte, es decir, con las muestras más cercanas al hiperplano.

Dependiendo de la situación:

- En **clasificación binaria**, intenta separar las clases con una **línea (2D)**, un **plano (3D)** o un **hiperplano (nD)**.
- En **problemas no lineales**, puede proyectar los datos a un espacio de mayor dimensión mediante **Kernels**.

La **función de decisión** se puede escribir como:

$$f(x) = w \cdot x + b$$

Donde:

- w es el vector de pesos (orientación del hiperplano),
- b es el sesgo (bias).
- El objetivo es **maximizar el margen** $\frac{2}{\|w\|}$ sujeto a:

$$y^{(i)} \left(w \cdot x^{(i)} + b \right) \geq 1 \quad \forall i$$

SVM es muy **eficaz en espacios de alta dimensión** y se desempeña bien cuando **existe un margen claro** entre las clases. Sin embargo, puede **no ser tan eficiente en datasets muy grandes**, es **sensible a la escala de los datos** (requiriendo normalización) y necesita un **ajuste cuidadoso de los hiperparámetros**, especialmente del Kernel y sus parámetros.

Consta de los siguientes Kernels más comunes:

- **Lineal:** Datos separables linealmente.
- **RBF (Radial Basis Function):** Para fronteras no lineales.
- **Polinómico:** Transforma con potencias del input.
- **Sigmoide:** Inspirado en redes neuronales.

A nosotros nos interesa los 2 primeros, *Lineal y RBF*.

Perceptrón Multicapa (*MLP*)

Un **MLP (Perceptrón Multicapa)**, en vez que todas las anteriores, es un tipo de **red neuronal *feedforward***, que consta de una capa de entrada, una o más capas ocultas y una capa de salida. **Cada neurona** de una capa está completamente **conectada con la siguiente**, y aplica una **función de activación** (por ejemplo, *ReLU* o *tanh*) para introducir no linealidades *(Figura 6.10)*[30].

Este modelo es capaz de **capturar relaciones complejas no lineales** y se **entrena mediante retropropagación del error (*backpropagation*)** utilizando optimizadores como *adam*, *sgd* o *lbfgs*.

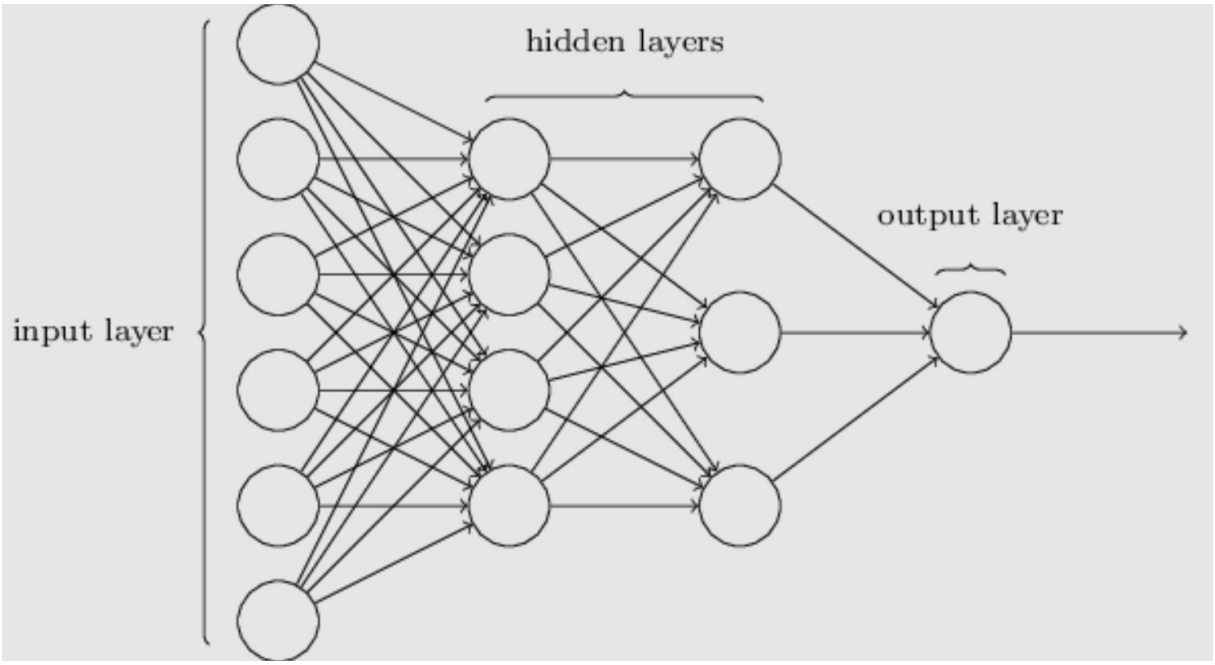

Figura 6.10: Diagrama representativo del funcionamiento del algoritmo MLP

Es ampliamente utilizado para **tareas de clasificación multiclase o regresión**, especialmente **cuando otras técnicas lineales no son suficientes**.

Ensemble por Votación

Los métodos **Ensemble combinan múltiples modelos individuales para mejorar el rendimiento general**, reducir el sobreajuste y aumentar la robustez de las predicciones.

El **modelo mediante votación combina predicciones de múltiples clasificadores base para tomar una decisión final**. La idea es que un conjunto de modelos

puede generalizar mejor que un único modelo por sí solo.

Existen dos tipos principales de votación:

- **Votación dura (hard):** Se elige la clase que obtiene **más votos (predicción de clase directa)** *(Figura 6.11)*[7].

- **Votación blanda (soft):** Se **promedian las probabilidades predichas** por cada clasificador, y se escoge **la clase con mayor probabilidad promedio** (solo válido si los clasificadores base pueden predecir probabilidades).

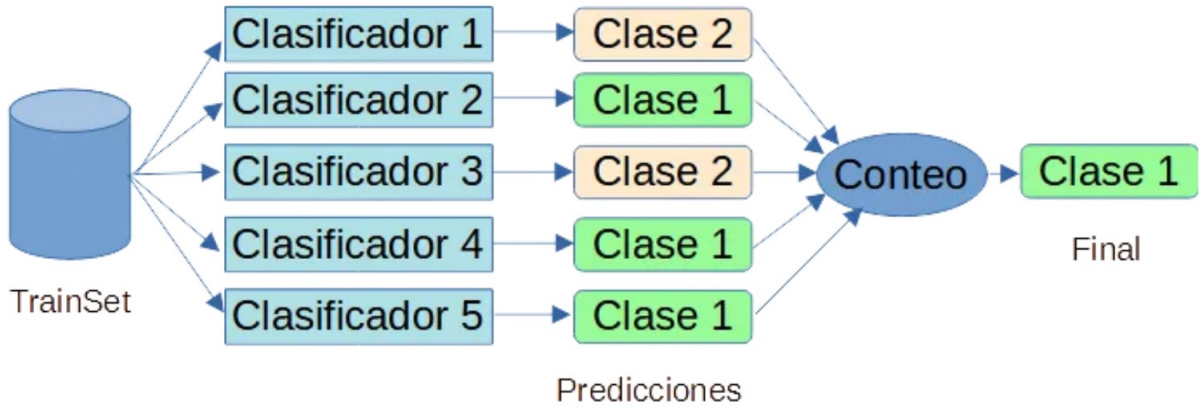

Figura 6.11: Diagrama representativo del funcionamiento del Ensemble por Votación (hard)

Este método es especialmente útil **cuando los clasificadores individuales tienen un rendimiento aceptable pero diverso**, y **no están altamente correlacionados** entre sí.

Aumento de Gradiente: Gradient Boosting

Gradient Boosting es un modelo de entrenamiento supervisado **basada en ensambles**, donde **múltiples modelos débiles** (habitualmente árboles de decisión) **se combinan de forma secuencial** para formar un modelo más robusto y preciso.

Su funcionamiento se basa en la idea de **minimizar una función de pérdida a través del método del gradiente descendente, corrigiendo iterativamente** los errores cometidos por los modelos anteriores *(Figura 6.12)*.

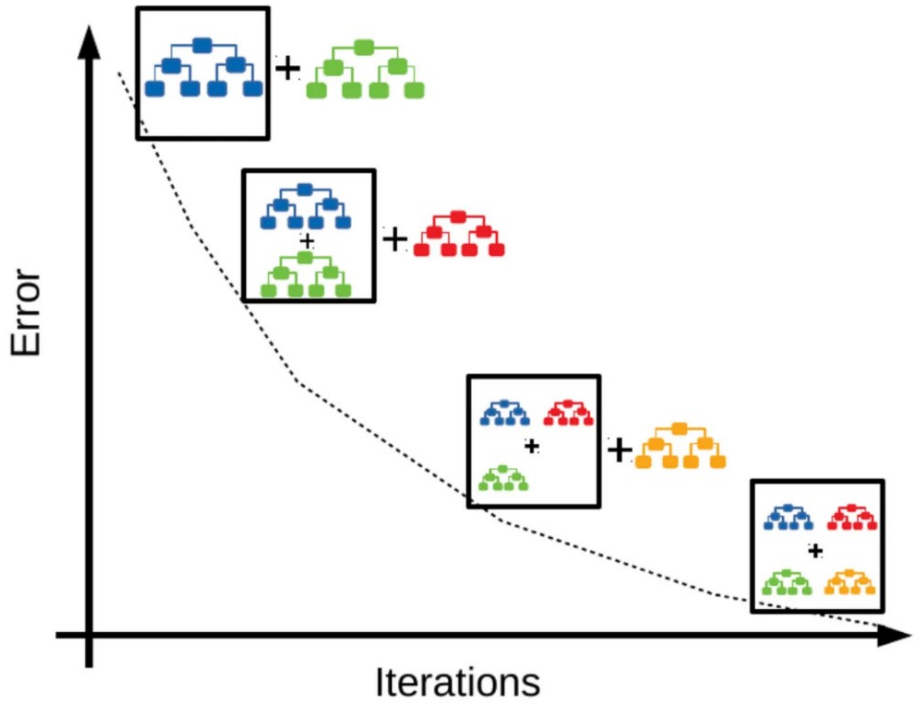

Figura 6.12: Diagrama representativo del funcionamiento del Aumento de Gradiente (GB)

Este algoritmo se implementó únicamente en *BlooDAI*. Esto fue así ya que durante su desarrollo se vio que los algoritmos como *Árboles de Decisión* y *Random Forest* eran los que **mejores resultados ofrecían para la predicción del diagnóstico presuntivo** del paciente animal. Al basarse este modelo en ellos, se consideró como buen algoritmo añadido a analizar.

6.4.3. Técnicas de Estandarización y Optimización en los Algoritmos

Durante toda la sección de algoritmos, muchas veces se ha mencionado tanto la **"escalabilidad"** de los datos como el **ajuste de los hiperparámetros**. Esto es debido a que **cada algoritmo trabaja con unas bases distintas y en cada una de ellas se debe ajustar** según el dataset administrado y a la labor que deberán cumplir su entrenamiento y pruebas.

Por lo tanto, esta sección detallará las **técnicas** usadas para sacar el mayor provecho de cada algoritmo.

Preprocesamiento de los Datos: Normalización o Estandarización

Algunos de los algoritmos que hemos usado, *SVM*, *Regresión Logística*, *KNN* y entre otros más, tienen **mejor funcionamiento si los datos se encuentran en la misma escala**.

Por ejemplo, si hay una característica con valores muy grandes y otra muy pequeños, erróneamente el algoritmo en cuestión **puede dar más importancia o peso a la de mayor valor** al afectar más a la **distancia entre puntos**.

En este caso, hemos usado ***StandardScaler***[18], una clase que ofrece *scikit-learn* para **escalar las características de un conjunto de datos**. Su objetivo es **transformar las variables a una escala estándar**, de manera que tengan **media cero y desviación estándar uno**.

Por lo tanto:

$$z = \frac{x - \mu}{\sigma}$$

Donde:

- x es el valor original de la característica.
- μ es la media de la característica.
- σ es la desviación estándar de la característica.
- z es el valor escalado.

Es decir, ***StandardScaler*** **resta la media y divide por la desviación estándar** para normalizar los datos.

Esto entonces hace que:

1. **Mejore el rendimiento de los modelos.**
2. **Mejore la convergencia en algoritmos de optimización.**
3. **Asegure una evaluación más justa.**

La siguiente **demostración** como ejemplo se encuentra incluida en el proyecto durante el desarrollo de *BlooDAI*, del cual el algoritmo de KNN se notó **una gran mejoría no sólo en precisión sino también en generalización** al aplicar este preprocesamiento en los datos *(Cuadro 6.3)*.

Métrica	KNN sin escalar (k=6)	KNN con StandardScaler (k=4)
Precisión Train	45.96 %	58.39 %
Precisión Test	31.43 %	35.71 %
F1-score Prom. Macro	0.22	0.32
F1-score Prom. Ponderado	0.27	0.33

Cuadro 6.3: Comparación de métricas de KNN con y sin escalar

A continuación procedemos con la técnica que nos ayudará a definir a cada algoritmo de la manera más óptima posible a partir de los hiperparámetros que están ligados a ellos.

Optimización de los algoritmos: Hiperparámetros

Cuando se trabaja con un modelo u algoritmo, se puede utilizar de la manera predeterminada usándolo directamente sin dar ninguna especificación de la forma en la que debe trabajar, y para datasets sencillos puede ser hasta suficiente. Sin embargo, **no es para nada práctico**.

Especificar correctamente los hiperparámetros de un modelo de aprendizaje automático es **fundamental**, ya que estos **controlan el comportamiento del algoritmo durante el entrenamiento**, afectando directamente su **capacidad para aprender patrones útiles, evitar errores comunes y generalizar bien** ante datos nuevos.

Esto también **previene**:

- **Sobreajuste (*overfitting*):** El modelo **aprende demasiado bien** los datos de entrenamiento y falla en los datos nuevos.

- **Subajuste (*underfitting*):** El modelo **no captura la complejidad** del problema y rinde mal incluso en entrenamiento.

Si no se hace correctamente, el modelo **podrá tener un rendimiento pobre** aun si es adecuado, **tardar más en entrenarse**, y **generar resultados poco confiables**.

Algunos hiperparámetros de ejemplo *(Cuadro 6.4)* que debemos considerar en algunos de los modelos que hemos utilizado serían:

Modelo	Hiperparámetro(s)	Descripción breve
KNeighborsClassifier	*n_neighbors*	Número de vecinos a considerar (K)
LogisticRegression	C	Inverso de la regularización; controla el sobreajuste
SVM	*kernel, C, gamma*	Tipo de núcleo y parámetros que afectan la flexibilidad
DecisionTreeClassifier	*max_depth, min_samples_split*	Controlan la complejidad del árbol
RandomForestClassifier	*n_estimators*	Número de árboles en el bosque
MLPClassifier (red neuronal)	*hidden_layer_sizes, learning_rate_init*	Tamaño de capas y tasa de aprendizaje

Cuadro 6.4: Hiperparámetros comunes en modelos de *scikit-learn*.

En nuestro caso, se usó **GridSearchCV**[12]. Ésta es una **técnica de búsqueda exhaustiva** que los ajusta evaluando todas las combinaciones indicadas en un conjunto de parámetros. Aplica *validación cruzada*, del cual profundizaremos más adelante.

Como siempre, vayamos a un caso de **ejemplo**, y en este caso lo haremos en ***BlooDAI*** que es la que requería de mayor necesidad de esta técnica.

```
from sklearn.model_selection import GridSearchCV

param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [6, 8, 12],
    'min_samples_split': [5, 10, 15],
    'class_weight': ['balanced', 'balanced_subsample'],
}

grid_search = GridSearchCV(
estimator=RandomForestClassifier(random_state=42),
param_grid=param_grid, cv=10, n_jobs=-1, verbose=2)
grid_search.fit(trainX, trainY)
print("Mejores parámetros encontrados:", grid_search.best_params_)

----
```

```
Fitting 10 folds for each of 54 candidates, totalling 540 fits
Mejores parámetros encontrados: {'class_weight': 'balanced_subsample',
'max_depth': 8, 'min_samples_split': 5, 'n_estimators': 200}
```

En este caso se aplica para ***Random Forest***, y a partir de los hiperparámetros que hemos delimitado aplica validación cruzada, lo que hace que *por cada 10 subconjuntos del dataset* se entrene con las *54 combinaciones posibles* de hiperparámetros, siendo un total de ***540 entrenamientos***.

Si comparásemos ahora con el modelo que habíamos entrenado sin determinar previamente los mejores hiperparámetros con los de ahora, en este caso de *Random Forest*:

Métrica	**Random Forest**	**Random Forest Optimizado**
Precisión Train	99.38 %	91.93 %
Precisión Test	74.29 %	74.29 %
F1-score Prom. Macro	0.72	0.75
F1-score Prom. Ponderado	0.73	0.72

Cuadro 6.5: Comparación de métricas entre Random Forest básico y optimizado.

Vemos que no sólo **generaliza mejor** entre las clases, sino que **evita el sobreajuste** que percibimos por el modelo básico, **sin perder precisión** *(Cuadro 6.5)*.

6.4.4. Regiones de cada Clasificador: Sobreajuste y Subajuste

Dado al bajo volumen de parámetros a tener en cuenta, en *SpecAI* se pudo realizar una **visualización de las clasificaciones** a partir de 2 parámetros (*MCH* y *RDW*) para poder **detectar** mejor casos de **sobreajuste o subajuste** *(Figura 6.13)*. Nos ayudará para escoger el mejor clasificador en esa situación.

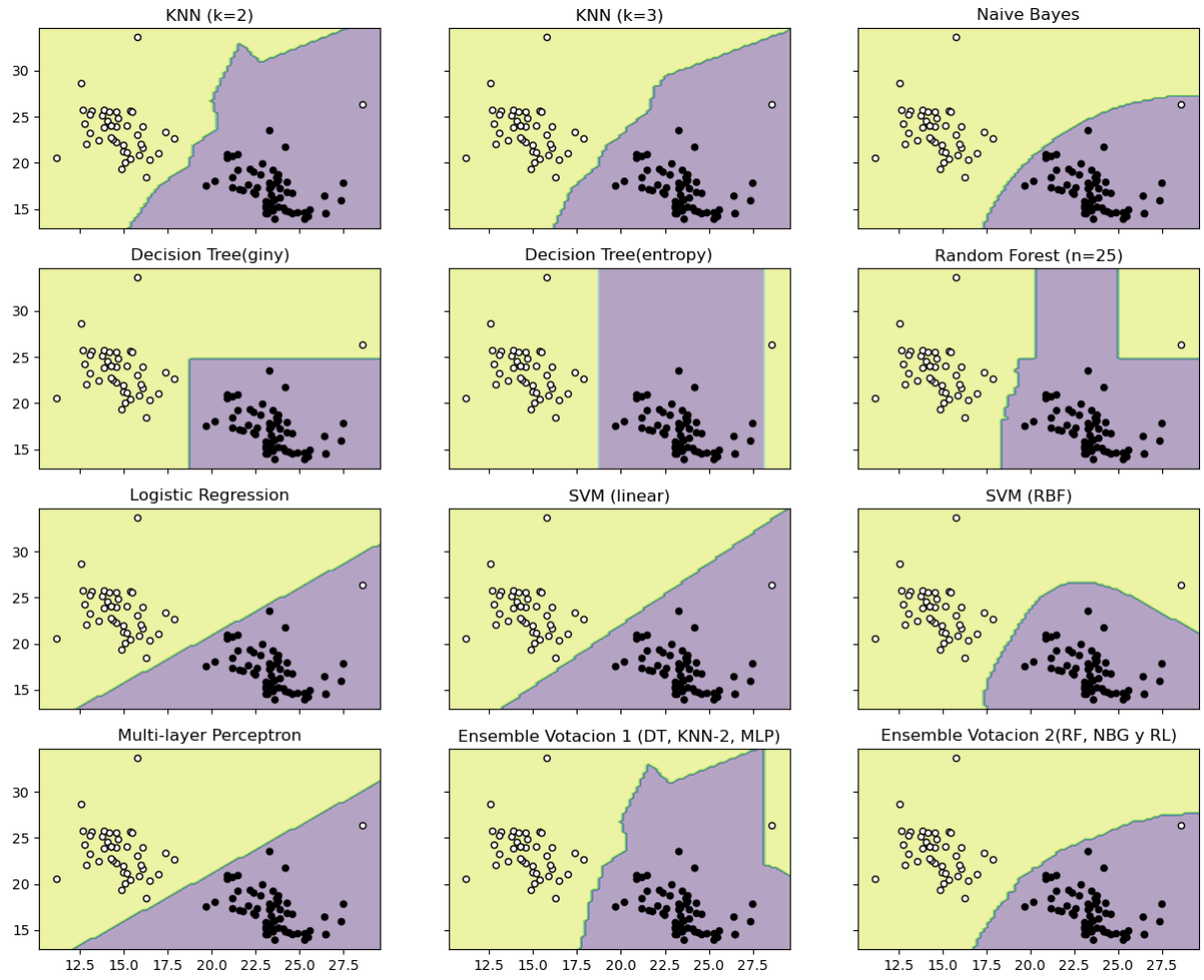

Figura 6.13: Regiones de cada Clasificador Destacable (SpecAI)

En esta imagen podemos destacar 3 cosas: delimitaciones **lineales con un crecimiento constante** (*Regresión Logística, SVM (Linear), MLP*), delimitaciones **abstractas** (*KNN k=2, Random Forest, Votación 1*), y delimitaciones más **acotadas y cerradas** (*Naive Bayes, SVM (RBF), Votación 2*).

Las que muestran un **mejor desempeño** son aquellos que son **lineales**, mientras que el resto muestran señales de **sobreajuste (cerradas)** o **subajuste (abstractas)**.

6.4.5. Validación Cruzada

Hay que recordar que anteriormente hemos separado en conjuntos de entrenamiento y pruebas, pero eso no nos **garantiza su desempeño real**. La forma para poder simularlo lo máximo posible es realizar una **validación cruzada**.

Lo que hace esta validación, brevemente comentada en la selección de hiperparámetros *(Sección 6.4.3)*, es dividir aún más el dataset en **varios conjuntos de datos (*folds*)**, de tal forma que se entrene con algunos de ellos y después se pruebe con el restante, repitiéndose el proceso varias veces. Es decir:

1. **Divides** el conjunto de datos en **k** subconjuntos o *folds*.
2. Para **cada iteración**, el modelo **entrena en k-1** de los subconjuntos y **evalúa su desempeño en el subconjunto restante**.
3. Este proceso **se repite k veces**, cada vez utilizando un *fold* diferente para la evaluación y los demás para el entrenamiento.

Ésto nos ayudará a tomar la decisión final del mejor modelo para la labor que necesitamos.

En el caso de *SpecAI* se hizo con **15** folds y *BlooDAI* con **10**. Esto es porque **la división deberá corresponderse según al volumen de datos y clases posibles** para que el desempeño no sea injustamente perjudicado por un elevado número de subconjuntos, no permitiéndole realmente **aprender patrones útiles de razonamiento**.

Esto significa que los entrenamientos realizados en cada modelo serán el doble de los subconjuntos (entrenamiento y prueba). Haciendo un total de **390 entrenamientos finales en SpecAI** y de **260 entrenamientos finales** en BlooDAI.

A continuación se dejarán reflejados los **resultados finales obtenidos** en esa validación cruzada en los mejores modelos que hemos ido detectando durante los entrenamientos.

Modelo	**Train Accuracy**	**Train StdDev**	**Test Accuracy**	**Test StdDev**
KNN (k=1)	0.980952	0.0486	0.955556	0.1133
KNN (k=2)	0.980952	0.0486	0.977778	0.0831
KNN (k=3)	0.980952	0.0486	0.977778	0.0831
Naive Bayes	0.980952	0.0486	0.933333	0.1333
Decision Tree (gini)	0.990476	0.0356	0.955556	0.1133
Decision Tree (entropy)	0.990476	0.0356	0.955556	0.1133
Random Forest (n=25)	0.990476	0.0356	0.955556	0.1133
Logistic Regression	0.971429	0.0571	0.977778	0.0831
SVM (linear)	0.971429	0.0571	0.977778	0.0831
SVM (RBF)	0.980952	0.0486	0.977778	0.0831
Multi-layer Perceptron	0.942857	0.1457	0.911111	0.1474
Ensemble Votación 1 (DT, KNN-2, MLP)	0.990476	0.0356	0.955556	0.1133
Ensemble Votación 2 (RF, NB, LR)	0.980952	0.0486	0.955556	0.1133

Cuadro 6.6: Resultados de la **validación cruzada** en *SpecAI*.

A partir de la validación cruzada, obtenemos la **media de precisión** tanto en entrenamiento como en pruebas, y la **varianza** que han llegado a alcanzar durante su validación del modelo en cuestión en ambas *(Cuadro 6.6)*.

Los más destacados para el modelo final ***SpecAI*** tras todo el análisis general serían:

- **K-Vecinos (K=2,3)**
- **SVM (Linear)**
- **SVM (RBF)**

Modelo	**Train Accuracy**	**Train StdDev**	**Test Accuracy**	**Test StdDev**
KNN (k=4) Scaled	0.260662	0.0820	0.200000	0.1457
KNN (k=6)	0.261029	0.0550	0.214286	0.1152
Naive Bayes	0.397426	0.0627	0.185714	0.1436
Decision Tree (Gini)	0.589706	0.1265	0.328571	0.2025
Decision Tree (Entropy)	0.459926	0.0859	0.228571	0.1457
Random Forest	0.701838	0.0466	0.585714	0.1491
Logistic Regression	0.422426	0.1075	0.385714	0.0915
SVM (Linear)	0.454044	0.0909	0.385714	0.0915
SVM (RBF)	0.310294	0.0956	0.271429	0.1187
Ensemble Votación 1 (RF, DT(Entropy), SVM-Linear)	0.466176	0.0907	0.228571	0.1457
Ensemble Votación 2 (RF, LR, SVM-Linear)	0.527574	0.0816	0.385714	0.0915
Ensemble Votación 3 (RF, DT(Gini), SVM-Linear)	0.577941	0.1434	0.414286	0.2429
Gradient Boosting	0.616176	0.1108	0.457143	0.1539

Cuadro 6.7: Resultados de la **validación cruzada** en *SpecAI*.

Los más destacados para el modelo final ***BlooDAI*** tras todo el análisis general *(Cuadro 6.7)* serían:

- **Random Forest**

- **Gradient Boosting**

6.5. SpecAI y BlooDAI

Tras todo el proceso anterior, se vieron los **mejores modelos** *(Cuadro 6.8)* previamente personalizados, siendo así:

IA	Modelo	Precisión (*Train*)	Desviación (*Train*)	Precisión (Test)	Desviación (*Test*)
SpecAI	SVM (Linear)	97,1 %	5,7 %	97,7 %	8,3 %
BlooDAI	Random Forest	70,1 %	4,6 %	58,6 %	14,9 %

Cuadro 6.8: SpecAI y BlooDAI

En el caso de **SpecAI** tuvimos un excelente resultado, siendo **SVM (Linear)** el más adecuado debido a su buena generalización y robustez al ruido. Esto nos **garantiza su producción**.

Por el contrario, la dificultad se haya en **BlooDAI**. Gracias a las particularidades que tiene el algoritmo de **Random Forest** (considerado el mejor clasificador para conjuntos de datos con muchas características, como es el caso) se alcanzó a una fiabilidad positiva a pesar de los obstáculos, pero la **escasez de casos más variados** (incluso tras la ligera ampliación) le pesa y no es adecuado en un uso real por su **leve generalización**.

6.6. Detalles en Implementación

Al utilizar Jupyter Notebook, nos permite segmentar todo el código de desarrollo para incluir información relevante mediante lenguaje ***Markdown***. Por lo tanto, para una mayor **legibilidad y usabilidad del proyecto**, toda relación con el desarrollo técnico asociado vendrá incluida con mayor profundidad en sus ficheros correspondientes y paso a paso.

6.7. Mejoras Futuras

Aunque hemos usado una par de técnicas para optimizar no solo los algoritmos sino también para tratar los datos, se puede haber usado alguna **técnica de sobre-muestro como SMOTE**[5].

SMOTE (*Synthetic Minority Over-sampling Technique*) es una técnica de sobre-muestreo usada para **balancear conjuntos de datos desequilibrados**. En lugar de duplicar ejemplos de la clase minoritaria, *SMOTE* genera nuevas instancias sintéticas

interpolando entre ejemplos reales cercanos de esa clase. Esta técnica **evitará la tendencia a foverecer a clases mayoritarias sin perder la diversidad** de los datos.

También utilizar **otros algoritmos de entrenamiento**, como por ejemplo ***XGBoost***[33].

Es un algoritmo de aprendizaje automático basado en árboles de decisión que forma parte de la **familia de métodos de *boosting***. Fue diseñado para ser altamente eficiente, flexible y portátil, y es especialmente **popular en competencias de ciencia de datos** por su gran rendimiento en tareas de clasificación y regresión. Se trata de una **versión mejorada del modelo *Gradient Boosting*** anteriormente visto, lo cual podría quizás ayudar a *BlooDAI* en específico.

Y finalmente, lo más importante realmente, **obtener más casos de pruebas reales**. Es el principal limitante de este proyecto en el diagnóstico presuntivo del animal.

Capítulo 7

Guía de Pruebas

El desarrollo de las IAs **SpecAI** y **BlooDAI** requieren de la búsqueda del mejor modelo mediante un **aprendizaje supervisado**[**AprendizajeSupervisado**] para cumplir sus determinadas funciones, por lo que en sí todo el desarrollo es un conjunto de pruebas[1].

A la hora de querer realizar **pruebas desde el punto de vista como usuario final** (en todos los distintos diagnósticos) sería un verdadero reto para alguien no profesional, ya que si se muestra cierta anomalía en alguna característica sanguinaria también se verían afectadas en otras dependientes, dando posibilidad de ser otros diagnósticos, sin contar de que hemos detectado durante la extracción un total de **17 diagnósticos posibles** (siendo '*Otros*' un conjunto de diagnósticos poco comunes general por falta de casuísticas) .

Por lo tanto, a continuación se indican **los diagnósticos más relevantes** junto con los respectivos valores de los parámetros que se deberían alterar.

[1] Todo este desarrollo vendrá reflejado en sus respectivos archivos de desarrollo junto con el uso final del usuario en Jupyter Notebook.

7.1. Estado Normal en Felinos y Caninos

Estos son los **valores normales** de sangre en Felinos *(Cuadro 7.1)*.

Especie Felina			
Categoría	**Parámetro**	**Mínimo**	**Máximo**
Glóbulos Rojos	Eritrocitos	6,54	12,2
	HCT (Hematocrito)	30,3	52,3
	HGB (Hemoglobina)	9,8	16,2
	MCV	35,9	53,1
	MCH	11,8	17,3
	MCHC	28,1	35,8
	RDW	15	27
	%RETIC	—	—
	RETIC	3	50
	RET-HE	13,2	20,8
Glóbulos Blancos	Leucocitos	2,87	17,02
	%NEU	—	—
	%LYM	—	—
	%MONO	—	—
	%EOS	—	—
	%BASO	—	—
	NEU	2,3	10,29
	LYM	0,92	6,88
	MONO	0,05	0,67
	EOS	0,17	1,57
	BASO	0,01	0,26
Plaquetas	PLQ	151	600
	MPV	11,4	21,6
	PCT	0,17	0,86

Cuadro 7.1: Valores de referencia normales (mínimo y máximo) para parámetros hematológicos veterinarios de Felino/a

Y estos son los **valores normales** de sangre en Caninos *(Cuadro 7.2)*.

Especie Canina			
Categoría	**Parámetro**	**Mínimo**	**Máximo**
Glóbulos Rojos	Eritrocitos	5,65	8,87
	HCT (Hematocrito)	37,3	61,7
	HGB (Hemoglobina)	13,1	20,5
	MCV	61,6	73,5
	MCH	21,2	25,9
	MCHC	32	37,9
	RDW	13,6	21,7
	%RETIC	—	—
	RETIC	10	110
	RET-HE	22,3	29,6
Glóbulos Blancos	Leucocitos	5,05	16,76
	%NEU	—	—
	%LYM	—	—
	%MONO	—	—
	%EOS	—	—
	%BASO	—	—
	NEU	2,95	11,64
	LYM	1,05	5,1
	MONO	0,16	1,12
	EOS	0,06	1,23
	BASO	0	0,1
Plaquetas	PLQ	148	484
	MPV	8,7	13,2
	PCT	0,14	0,46
	PDW	9,1	19,4

Cuadro 7.2: Valores de referencia normales (mínimo y máximo) para parámetros hematológicos veterinarios de Canino/a

Ahora pasemos a los **estados anómalos más destacables** según la serie sanguínea en ambas especies.

7.2. Estados de Anemia

Las anemias son detectadas a partir de valores normalmente bajos en la **serie roja** de la sangre (glóbulos rojos) *(Figura 7.1)*[1].

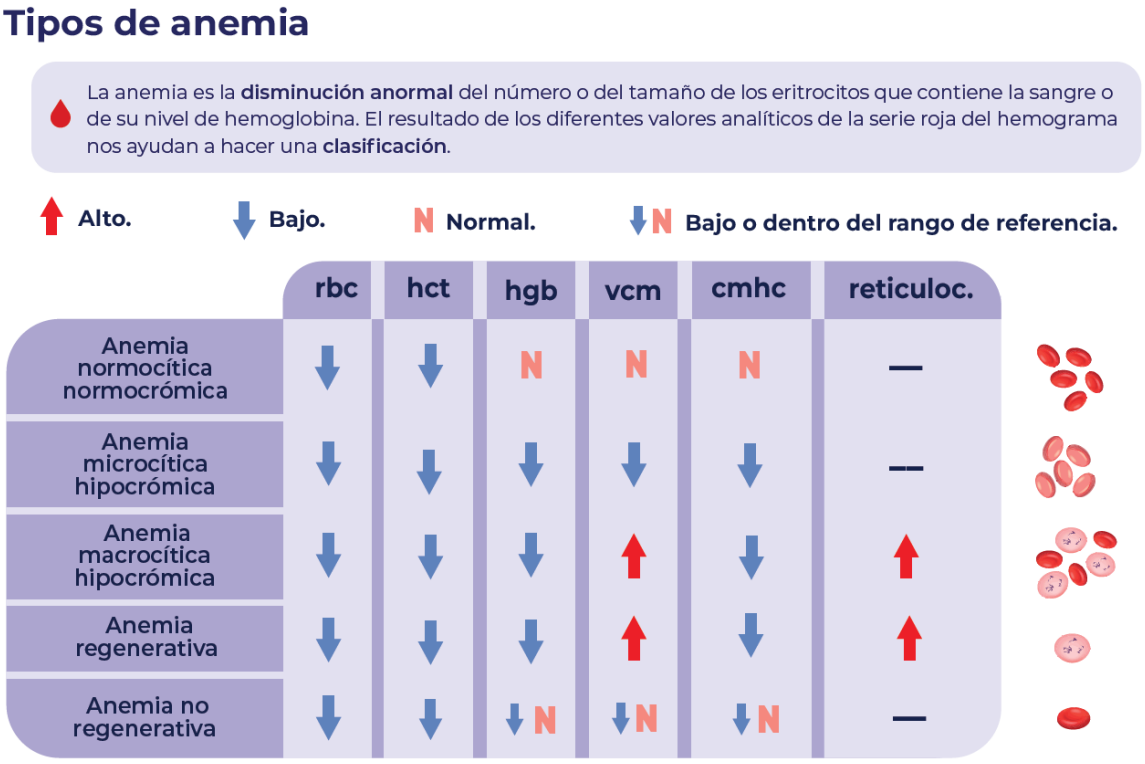

Figura 7.1: Registro de anemias mediante hemograma.

Una observación importante a tener en cuenta en estos diagnósticos es que las tres primeras anemias realmente son subcategorías de las **anemias regenerativa y no regenrativa**, por el cual **BlooDAI**[2] clasificará.

Además, en esta serie se puede detectar **Reticulocitosis** (***RETIC***) si los reticulocitos son elevados.

[1]Esta imagen viene facilitada en unos de los PDFs adjuntos al proyecto. Hay que tener en cuenta su desvarianza de valores normales al estar desactualizado con las máquinas actuales.

[2]Como se ha mencionado anteriormente, esta IA requiere de una ampliación de casos para poderse usar con una fiabilidad deseable en entornos reales.

7.3. Estados de Leucograma

Al contrario con las anemias, los leucogramas tienen ciertas diferencias y no se pueden tener subcategorías entre sí a partir de los valores de la **serie blanca** de la sangre (glóbulos blancos) *(Figura 7.2)*.

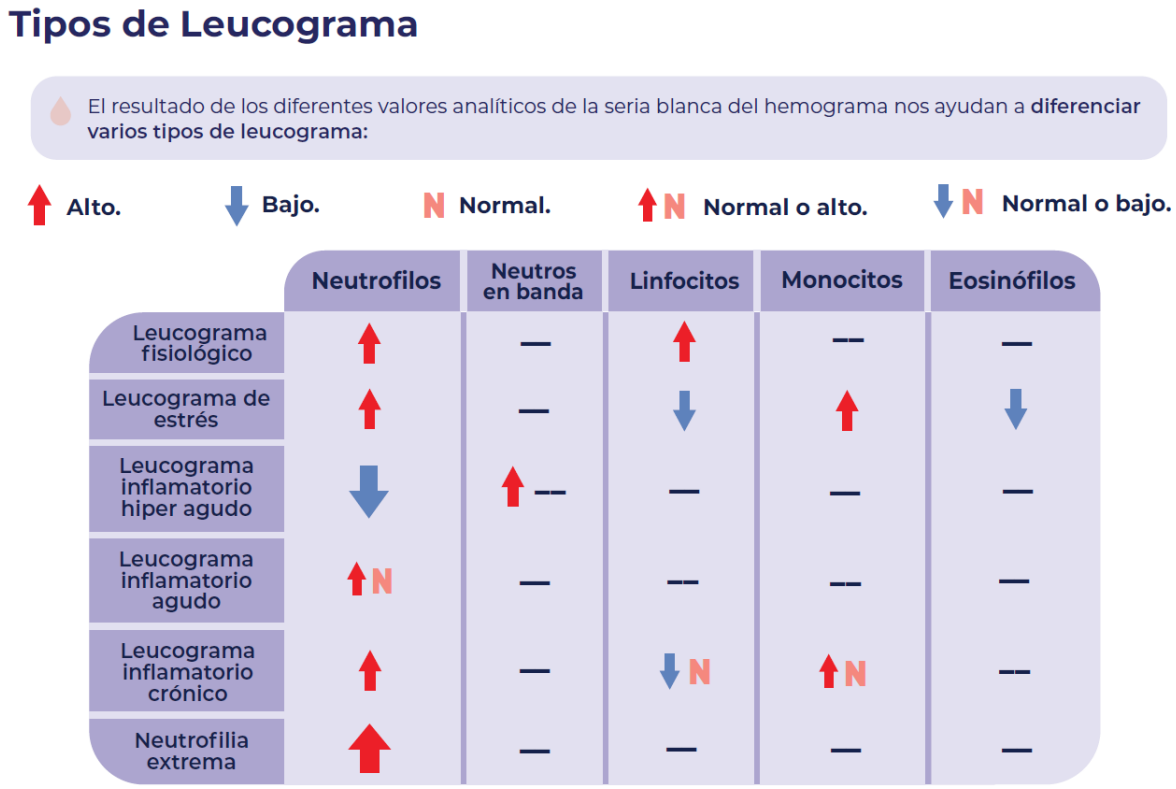

Figura 7.2: Registro de leucograma mediante hemograma.

Un detalle a tener en cuenta es que sólo se tienen contemplados en este proyecto (por los datos extraídos) los siguientes tipos:

- **Leucograma de estrés**
- **Leucograma inflamatorio agudo**
- **Leucograma inflamatorio hiper agudo**
- **Neutrofilia extrema**

Una mención honorífica de otras casuísticas serían las determinadas por el valor único de los **linfocitos** (***LYM***):

- **Linfopenia:** Valores bajos.
- **Linfocitosis:** Valores altos.

7.4. Estados Anómalos en las Plaquetas

Para finalizar, en el **conjunto de plaquetas** (***PLQ***) se detectan dos tipos de diagnósticos:

- **Trombocitosis:** Valores altos en plaquetas.
- **Trombocitopenia:** Valores bajos en plaquetas.

Capítulo 8

Conclusiones

En conclusión, hemos conseguido:

- Identificar las **pruebas más recurrentes y sus departamentos** donde se pueda **explorar otras iniciativas** para el sector.
- Obtener los **conocimientos** necesarios del sector y unos **datos iniciales** para proceder con los entrenamientos de ambos modelos de IA.
- Una IA que puede usarse para **determinar la especie** con datos hemográficos (**SpecAI**).
- Una IA que pueda **diagnosticar al paciente**, requiriendo de una pequeña ampliación de casos reales para evitar ser sesgada por los diagnósticos más comunes, (**BlooDAI**).
- Con posibilidad de la **entrada a producción del modelo *SpecAI***.

Sin embargo:

- ***BlooDAI* no se puede usar en producción**. Se requiere de **más datos y entrenamiento**.
- No se contempla aconsejar alguna receta médica para el diagnóstico, como se sugirió en el comunicado inicial.

8.1. Futuros Proyectos

Algunas **líneas de trabajo futuras o mejoras** que hemos identificado son las siguientes:

- Desarrollar una **herramienta software** de conexión para que se pueda usar mediante una interfaz amigable y sencilla, como **aplicación** para veterinarios o auxiliares.

- **Colaborar con IDEXX** para poder implementarlo en sus propias máquinas de hematología, evitando así la inserción de datos manual. Es la empresa que diseñó la maquinaria, y se podría hacer una propuesta.
- Probar con **otras especies** para ampliar el rango de aplicación del proyecto DACAI.

8.2. Reflexiones

Durante el desarrollo del trabajo surgieron numerosas **incertidumbres derivadas del desconocimiento** completo del sector. La **falta de experiencia previa**, más allá de la vivencia personal como dueña de animales de compañía, dificultó acercarse a la realidad del tema tratado. Buscar información en internet tampoco resultó una solución adecuada, ya que gran parte de los datos disponibles eran difíciles de interpretar o poco fiables, lo que aumentaba la sensación de inseguridad y limitaba la validez de la información recopilada.

Contactar con las clínicas veterinarias era sencillo, pero el hecho de conseguir una colaboración fue distinto. Se intentó con unas cuantas más de las 2 que finalmente colaboré (y un fuerte agradecimiento por éstas), incluso con la ***Facultad de Veterinaria de la Universidad Complutense de Madrid*** sin éxito. Una cosa a destacar al intentar contactar con la universidad es que una joven estudiante de doctorado de veterinaria estuvo interesada en mi proyecto para hacer una colaboración para su tesis, a la cual tuve que rechazar respetuosamente.

Hago un llamamiento en este caso a ***Clínica Veterinaria San Lorenzo***, quienes tuvieron una paciencia y comprensión increíbles, dejando aprender con libertad su forma de trabajar, los procesos que siguen y las herramientas que usan en sus propias instalaciones.

Debido a esto, aquí es donde comenzó la ligera desviación del alcance a lo que se había estipulado en la comunicación inicial al no incluir finalmente una recomendación de tratamiento. Llevarlo acabo habría supuesto una extensión de trabajo del cual, incluso con el conocimiento actual que se ha obtenido, resultaría complejo por buscar la forma de orientarlo.

Sin embargo, se ha desarrollado una IA dedicada a la predicción del tipo de animal a tratar, permitiendo una mayor escalabilidad y usabilidad con menores limitaciones.

La única **limitación final** ha sido la **falta de gran cantidad datos entre los distintos casos**. Por las condiciones, herramientas y formas de trabajar dentro de las clínicas, la extracción de datos se tuvo que realizar de manera manual, dedicando a sí mucho trabajo de extracción a cambio de evitar un tratamiento de datos más exhaustivo.

Parte III

Manuales de la Aplicación

Capítulo 9

Manual de Instalación

Recomendamos tener instalado **Anaconda Navigator**[1], que permitirá gestionar las descargas de paquetes extra sin tener que usar el panel de comandos, además de incluir la posibilidad de generarse nuevos entornos virtuales para adaptarlo a las recomendaciones del proyecto si se desea.

Anaconda Navigator: Página de descarga **aquí**.

Una vez instalado, lo primero que saldrá al ejecutarlo aparecerá el inicio *(Figura 9.1)*.

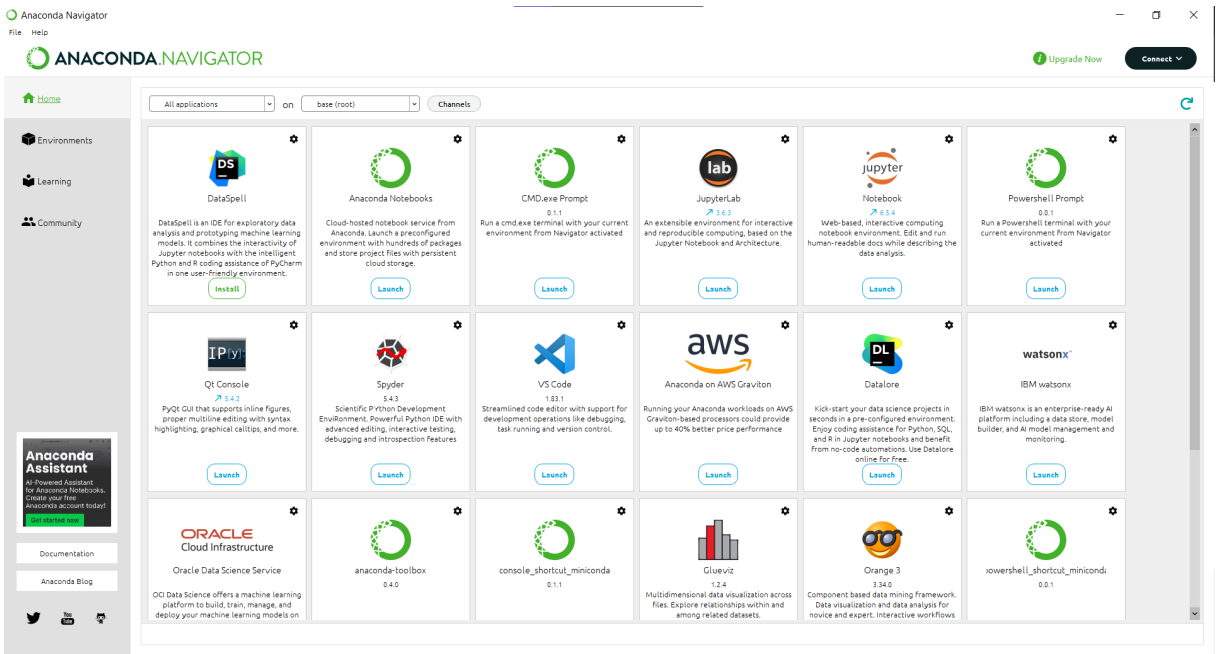

Figura 9.1: Inicio de Anaconda

[1]Tendrás además acceso a una zona de aprendizaje y de comunidad.

Para poder acceder a los archivos del proyecto, se cliqueará sobre "***Launch***"debajo de **Jupyter Notebook**. Te abrirá el árbol de directorios en el navegador predeterminado, y solo tendrás que manejarte por él hasta alcanzar al repositorio del trabajo.

Sin embargo, **previamente** será necesario tener descargados las librerías esenciales comentadas anteriormente en este documento, pudiendo gestionarlo a partir de la sección "***Environments***". Aquí podrás revisar los paquetes ya instalados, aquellos que no y respectivas versiones, y la posibilidad de gestionarlo mediante distintos entornos *(Figura 9.2)*.

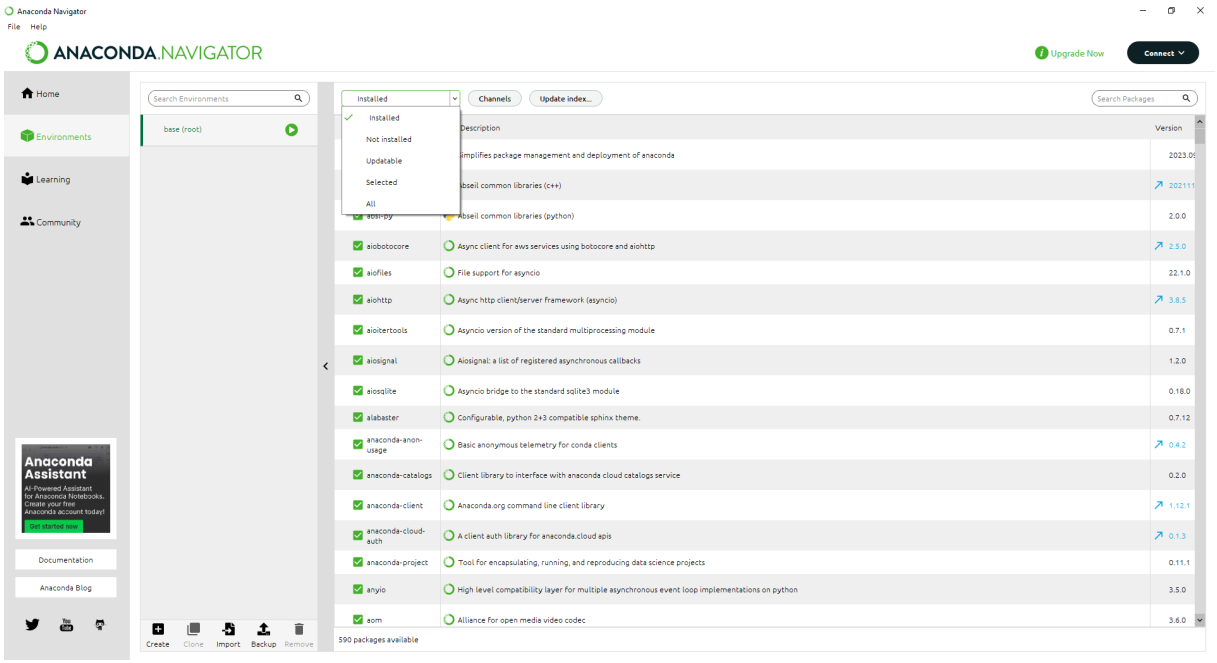

Figura 9.2: Gestión de paquetes y entornos mediante Anaconda Navigator

En caso de querer instalar librerías **mediante el terminal**, se deberá realizar de las siguientes maneras:

```
pip install <libreria>
pip install --force-reinstall '<libreria>==<version>'
```

Este terminal de comandos también se podrá abrir desde "*Environments*" al hacer click sobre el botón verde del entorno de interés, instalándose en ese.

Capítulo 10

Manual de Usuario

Este capítulo incluye una breve guía de uso para el usuario final como cualquier experto e interesado en el desarrollo detrás de las IAs junto con sus pruebas pero con una índole más técnica, denominado *Manual de Administración*.

10.1. Manual de Usuario

Una vez seguido los pasos del manual de instalación y encontrarse dentro del repositorio del proyecto, se deberá acceder al directorio ***1_JupyterNote***. Dentro de éste, se destacarán tres ficheros sobre el resto, el cual de interés exclusivo será el fichero ***DACAI***.

Se abrirá de manera automática en una pestaña aparte del navegador, y dentro vendrá los **pasos a seguir** para poder utilizar las IAs ya previamente incluidas, **SpecAI** y **BlooDAI**.

10.2. Manual de Administración

En el caso de un usuario más técnico y superior, dentro del repositorio principal del proyecto, accederá a la carpeta ***1_JupyterNote***. Dentro de ésta, tres ficheros serán de interés:

- **DACAI:** Fichero de ejecución y uso final de las dos IAs desarrolladas.
- **SpecAI_Development:** Fichero relacionado con el desarrollo y entrenamiento de la IA para detectar especies.
- **BlooDAI_Development:** Fichero relacionado con el desarrollo y entrenamiento de la IA para determinar el diagnóstico presuntivo del animal.

Todos estos ficheros contendrán una guía paso a paso y de aprendizaje del desarrollo y uso del proyecto, tanto técnica como funcionalmente.

Parte IV

Apéndices

Apéndice A

Contenido

Este proyecto contendrá la siguiente organización:

- La **memoria** del proyecto (este documento).
- Carpeta **0_Investigacion**.
- Carpeta **1_JupyterNote**.
- Carpeta **2_Documentacion**.

Dentro de la carpeta **0_Investigacion** se almacena toda la información recogida sobre el sector veterinario, en el que:

- **Consultas %:** Se reflejan todos las apuntes iniciales y de investigación final de las consultas recogidas en ese tiempo previo al desarrollo.
- **Procyte Dx:** Investigación inicial de los resultados hematológicos de las consultas.
- **Catalyst One:** Estudio realizado sobre los resultados bioquímicos antes de su previo descarte.

En **1_JupyterNote** se encontrará todo lo relacionado con el desarrollo del trabajo, en el que destacamos:

- **Datos % y Extended %:** Ficheros CSV con los casos reales para entrenamiento, junto con sus versiones XLS.
- **BlooDAI.pkl:** IA desarrollada para el diagnóstico presuntivo.
- **SpecAI.pkl:** IA desarrollada para determinar la especie.
- ***DACAI.ipynb*:** Fichero de Jupyter Notebook con un ejemplo de uso final del proyecto.

- ***BlooDAI_Development.ipynb*:** Fichero de Jupyter Notebook con el desarrollo, la guía y las pruebas de **BlooDAI**.
- ***SpecAI_Development.ipynb*:** Fichero de Jupyter Notebook con el desarrollo, la guía y las pruebas de **SpecAI**.

Para finalizar, el directorio **2_Documentacion** contendrá una ligera información extra desde un punto de vista veterinario, ya sea de los procedimientos o del entendimiento de los resultados de un análisis de sangre.

Referencias

[1] Moez All. *Machine Learning: Aprendizaje supervisado*. 2024. URL: https://www.datacamp.com/es/blog/supervised-machine-learning (visitado 13-05-2025).

[2] AntiguoRincon.com. *El origen histórico de las mascotas*. 2024. URL: https://www.antiguorincon.com/blog/el-origen-historico-de-las-mascotas/ (visitado 11-05-2025).

[3] Kent Beck, Mike Beedle y Arie van Bennekum. *Manifiesto Ágil*. 2001. URL: https://agilemanifesto.org/ (visitado 11-05-2025).

[4] Veterinaria Patitas Blue. *Exploración Física General en felinos*. 2022. URL: https://www.facebook.com/veterinariapatitasblue/posts/el-examen-f%C3%ADsico-o-cl%C3%ADnico-general-efg-permite-detectar-si-hay-alteraciones-en-l/1567247143645358/ (visitado 11-05-2025).

[5] Guillaume Lemaitre y Fernando Nogueira y Christos K. Aridas. *Técnica de Sobremuestreo: SMOTE*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (visitado 25-05-2025).

[6] Alto Servicios Comunicación. *Preguntas en una exploración física veterinaria*. 2020. URL: https://clinicaveterinariaalcazaba.com/preguntas-en-una-exploracion-fisica-veterinaria/ (visitado 11-05-2025).

[7] Nicolás Arrioja Landa COsio. 2022. URL: https://medium.com/@nicolasarrioja/gu%C3%ADa-definitiva-a-las-t%C3%A9cnicas-de-ensemble-7a4bb1203bcb (visitado 25-05-2025).

[8] Desconocido. *Los expertos advierten sobre ChatGPT y su uso en educación: no es infalible y puede mermar la memoria*. 2024. URL: https://www.tribunasalamanca.com/noticias/359833/los-expertos-advierten-sobre-chat-gpt-y-su-uso-en-educacion-no-es-infalible-y-puede-mermar-la-memoria (visitado 11-05-2025).

[9] Scikit learn Documentation Team. *Decission Tree Classifier — scikit-learn documentation*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html (visitado 24-05-2025).

[10] Scikit learn Documentation Team. *Ensemble: Gradient Boosting — scikit-learn documentation*. URL: https://scikit-learn.org/stable/modules/ensemble.html# (visitado 24-05-2025).

[11] Scikit learn Documentation Team. *Ensemble: Voting Classifier — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/ensemble.html#voting-classifier (visitado 24-05-2025).

[12] Scikit learn Documentation Team. *GridSearchCV — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (visitado 24-05-2025).

[13] Scikit learn Documentation Team. *Gussian Naive Bayes — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html (visitado 24-05-2025).

[14] Scikit learn Documentation Team. *K-Neighboors Classifier — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (visitado 24-05-2025).

[15] Scikit learn Documentation Team. *Logistic Regression — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (visitado 24-05-2025).

[16] Scikit learn Documentation Team. *MLP Classifier — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html (visitado 24-05-2025).

[17] Scikit learn Documentation Team. *Random Forest Classifier — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (visitado 24-05-2025).

[18] Scikit learn Documentation Team. *StandardScaler — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html (visitado 24-05-2025).

[19] Scikit learn Documentation Team. *SVM Classification — scikit-learn documentation.* URL: https://scikit-learn.org/stable/modules/svm.html#svm-classification (visitado 24-05-2025).

[20] Autónomos y Emprendedores. 2025. URL: https://www.autonomosyemprendedor.es/articulo/noticias-de-empresa/cepyme-asegura-que-inflacion-fuerza-empresas-exigir-antes-cobro-facturas/20230524132915030599.html.

[21] Instituto de investigación Punto de Fuga. "Estudio Él nunca lo haría". En: *Fundación Affinity* (2024). URL: https://static.fundacion-affinity.org/cdn/farfuture/Yy4fGT4TV-fJUGY9fCJKWGqvBOyRmMqCPlYCTcEvVcI/mtime:1718631992/sites/default/files/whitepaper-abandono-2024.pdf (visitado 11-05-2025).

[22] M Trigás Gallego. "Metodología scrum". En: *Universitat Oberta de Catalunya* (2012). (Visitado 11-05-2025).

[23] Inc IDEXX Laboratories. *ProCyte Dx: Analizador de hematología.* 2021. URL: https://www.idexx.es/files/procyte-dx-operators-guide-es-es.pdf (visitado 13-05-2025).

[24] Gaspar González Jurado-Gutiérrez. *El auge de la IA: una mirada al pasado y presente de la tecnología*. 2024. URL: https://www.telefonica.com/es/sala-comunicacion/blog/auge-ia-mirada-al-pasado-presente-tecnologia/ (visitado 11-05-2025).

[25] Laura Castellano Lendínez. "Kanban. Metodología para aumentar la eficiencia de los procesos". En: *3c Tecnología: glosas de innovación aplicadas a la pyme* 8.1 (2019), págs. 30-41. (Visitado 11-05-2025).

[26] Inés López. *Los animales de compañía alivian la soledad en los jóvenes: "Los consideran sus aliados"*. 2024. URL: https://www.20minutos.es/noticia/5535426/0/los-animales-compania-alivian-la-soledad-en-los-jovenes/ (visitado 11-05-2025).

[27] David Mouriquand. *ChatGPT cumple un año: la inteligencia artificial que amenaza a la cultura y la creatividad*. 2023. URL: https://es.euronews.com/cultura/2023/11/30/chatgpt-cumple-un-ano-la-inteligencia-artificial-que-amenaza-a-la-cultura-y-la-creatividad (visitado 11-05-2025).

[28] itop Tecnología y Negocio. *Scrikit-Learn*. URL: https://www.itop.es/soluciones-tecnologicas/business-analytics-business-intelligence/scikit-learn.html (visitado 11-05-2025).

[29] OpenAI. *ChatGPT*. URL: https://openai.com/es-ES/index/chatgpt/ (visitado 26-05-2025).

[30] rcassani. 2017. URL: https://github.com/rcassani/mlp-example?tab=readme-ov-file (visitado 25-05-2025).

[31] S.J. Russell, P. Norvig y J.M.C. Rodríguez. *Inteligencia artificial: un enfoque moderno*. Colección de Inteligencia Artificial de Prentice Hall. Pearson Educación, 2004. ISBN: 9788420540030. URL: https://books.google.es/books?id=yZCVPwAACAAJ (visitado 11-05-2025).

[32] Juan Ruiz Sierra. *Perros frente a niños: cómo la covid hizo bajar la natalidad pero animó las adopciones de mascotas*. 2022. URL: https://www.epe.es/es/sociedad/20220618/perros-frente-ninos-covid-13883158 (visitado 11-05-2025).

[33] Simplilearn. *XGBoost*. 2025. URL: https://www.simplilearn.com/what-is-xgboost-algorithm-in-machine-learning-article (visitado 25-05-2025).

[34] José Sáez. *¿Qué es un CRM? ¿Cómo funciona?* 2024. URL: https://inforges.es/blog/que-es-un-crm-y-como-funciona/ (visitado 13-05-2025).

[35] Academia ZBrain Veterinaria. *Por qué la Exploración Física Veterinaria es Clave*. 2024. URL: https://veterinaria.zbrain.es/la-exploracion-fisica-veterinaria-es-clave/ (visitado 11-05-2025).