SISTEMA PARA LA ESTIMACIÓN DE PESO DE GANADO BOVINO A PARTIR DE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

Miller Eduardo Perdomo Trujillo
Código: 9014842

Proyecto Aplicado para optar al título de
Magister en Ciencia de Datos

Director(a)
PhD. Diego Luis Linares Ospina
CO-DIRECTOR:
PhD. Gloria I. Álvarez

FACULTAD DE INGENIERÍA Y CIENCIAS
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI, DICIEMBRE DE 2025

FICHA RESUMEN

TÍTULO DEL PROYECTO: SISTEMA PARA LA ESTIMACIÓN DE PESO DE GANADO BOVINO A PARTIR DE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

1. ÁREA DE TRABAJO: INDUSTRIAL
2. TIPO DE PROYECTO (Aplicado, Innovación, Investigación): APLICADO
3. ESTUDIANTE(S): Miller Eduardo Perdomo Trujillo
4. CORREO ELECTRÓNICO: millerperdomo@javerianacali.edu.co
5. DIRECCIÓN Y TELEFONO: Calle 45 #8ª-25 Cali-Colombia; 3157658601
6. DIRECTOR: PhD. Diego Luis Linares
7. VINCULACIÓN DEL DIRECTOR: Profesor Universidad Javeriana Cali
8. CORREO ELECTRÓNICO DEL DIRECTOR: dlinares@javeriana.edu.co
9. CO-DIRECTOR (Si aplica): PhD. Gloria I. Álvarez; Correo: galvarez@javeriancali.edu.co
10. GRUPO O EMPRESA QUE LO AVALA (Si aplica):
11. OTROS GRUPOS O EMPRESAS:
12. PALABRAS CLAVE (al menos 5): Engrasamiento, Machine Learning, CNN, TensorFLow, Keras, Colab, features.
13. FECHA DE INICIO: 7 de enero del 2025
14. DURACIÓN ESTIMADA (En meses): 10 Meses

RESUMEN:

La ganadería bovina requiere métodos precisos, prácticos y económicos para estimar el peso vivo, un parámetro esencial para el manejo productivo, la salud y la comercialización. Los métodos tradicionales básculas y cintas métricas pueden resultar costosos, poco accesibles en zonas rurales y generar estrés en los animales.

Este proyecto desarrolló un sistema de estimación de peso mediante visión artificial y aprendizaje automático, optimizado para dispositivos móviles sin conexión a internet. Se construyó un conjunto de datos compuesto por 513 registros tabulares y 17 899 imágenes de vista posterior, permitiendo integrar información morfométrica y visual en distintos enfoques de modelado.

Se evaluaron dos líneas metodológicas principales. La primera utilizó modelos de regresión sobre datos tabulares (SVR, Random Forest, XGBoost y MLP). Los modelos de conjunto lograron los mejores resultados, destacando XGBoost, que alcanzó un $R^2 > 0.99$ y un MAE de 3.27 kg. Este desempeño confirma que las variables morfométricas permiten una estimación altamente precisa del peso vivo.

El segundo enfoque exploró modelos multimodales e híbridos basados en imágenes. El mejor resultado provino de una arquitectura paralela que combina una ResNet50 preentrenada para extraer características visuales con un MLP Regressor para procesar los datos tabulares. Esta fusión alcanzó un $R^2 = 0.74$ y un MAE = 21.57 kg. Es relevante subrayar que estas métricas corresponden al modelo híbrido y no a uno puramente visual. Al evaluar únicamente la rama de visión, el desempeño disminuyó ($R^2 \approx 0.60$ y MAE $\approx$ 29.9 kg), lo que evidencia que la información tabular sigue siendo esencial para obtener predicciones confiables y mejora significativamente la precisión final.

Además, se desarrolló un prototipo móvil Android utilizando TensorFlow Lite para realizar inferencias directamente en el dispositivo (Edge Computing). La aplicación puede estimar el peso a partir de una fotografía del tercio posterior del animal y demostró ser funcional en entornos rurales con conectividad limitada. No obstante, la precisión basada exclusivamente en imágenes aún se encuentra por debajo de la obtenida con datos morfométricos.

En conclusión, los datos tabulares ofrecen la mayor precisión disponible; la combinación multimodal mejora respecto a la visión sola, pero no supera al mejor modelo tabular (XGBoost). El prototipo móvil demuestra viabilidad práctica y potencial de aplicación real, aunque requiere mejoras en la integración de modalidades y en la robustez ante variabilidad de captura para alcanzar niveles comparables a los métodos tradicionales.

CONTENIDO

1. DEFINICIÓN DEL PROBLEMA 9
1.1. Planteamiento del problema 9
1.2. Formulación del problema 10
2. OBJETIVOS 11
2.1. Objetivo General 11
2.2. Objetivos Específicos 11
3. MARCO TEÓRICO Y ANTECEDENTES 12
3.1. Marco Teórico 12
3.1.1. Métodos de Estimación de Peso Vivo Bovino 12
3.1.2. Fundamentos de Aprendizaje Automático (Machine Learning) 12
3.1.3. Modelos de Regresión Continua 14
3.1.4. Redes Neuronales Artificiales 17
3.1.5. Aprendizaje por transferencia 19
3.1.6. Modelos Multimodal (Hibrido) 21
3.1.7. Validación Cruzada 21
3.1.8. Métricas de Evaluación de Modelos 22
3.1.9. Modelos ML/CNN en Móvil 23
3.2. Antecedentes 23
4. OBTENCIÓN Y PROCESAMIENTO DE LOS DATOS 27
4.1. Adquisición de los datos 27
4.2. Descripción general 27
4.2.1. Exploración de Datos Tabulares 28
4.2.2. Imágenes del Set de Datos 33
4.3. Procesamiento Inicial de Datos 34
4.3.1. Transformación de los Datos. 35
4.3.2. División de Datos 36
4.3.3. Data Set de Imágenes 36
5. MODELOS INICIALES 38
5.1. Regresiones Iniciales 38
5.1.1. Modelos de ML Tradicionales. 38
5.1.2. Perceptrón Multicapa – MLP 43
5.1.3. Estrategia de Validación (K-Fold-Cross Validation) 45
6. MODELOS MULTIMODALES (HÍBRIDOS) 47
6.1. Modelo Multimodal Paralelo. 47
6.2. Modelo Multimodal en Serie 49

6.2.1. Conjunto de datos 50
6.2.2. Preprocesamiento y pipeline de datos 51
6.2.3. Modelo de extracción de características: VGG16 con Fine-Tuning 51
6.2.4. Cabeza del modelo y función de pérdida 53
6.2.5. Extracción de embeddings. 53
6.2.6. Modelo de regresión basado en RandomForest 53
6.2.7. Evaluación del sistema 54
6.2.8. Integración del pipeline completo 54
7. ANÁLISIS DE RESULTADOS 55
7.1. Resultados 55
7.2 Análisis de Resultados 55
7.2.1. Resultados del Modelo Multimodal en Serie 57
8. LA APLICACIÓN MÓVIL PROTOTIPO 61
8.1 Tecnología y Arquitectura 61
8.1.1. Arquitectura de Despliegue del Modelo Híbrido 62
8.2. Flujo de Usuario y Funcionalidad 63
8.3. Evaluación Técnica y de Usabilidad del Prototipo 66
9. CONCLUSIONES Y TRABAJOS FUTUROS 68
9.1. CONCLUSIONES 68
9.2. TRABAJOS FUTUROS 69
10. REFERENCIAS BIBLIOGRÁFICAS 71
Bibliografía 71

LISTA DE FIGURAS

Figura 1: Arquitectura de una SVM de regresión 15
Figura 2: Arquitectura de Random Forest 16
Figura 3: Arquitectura de XGBoost Regresor. 17
Figura 4: Red neuronal perceptrón multicapa 18
Figura 5: Arquitectura de una CNN. 19
Figura 6: Arquitectura de la Resnet50 20
Figura 7: Arquitectura de la VGG16 20
Figura 8: a) Distribución de frecuencia de la variable Altura (height_in_inch). b) Distribución de frecuencia variable peso (weight_in _kg) 29
Figura 9: a) Mapa de color de correlación entre las variables numéricas. b) Relación entre la altura y peso del animal 31
Figura 10: a) Porcentaje de animales por razas. b) Distribución de por raza. 32
Figura 11: Distribución del peso según el tamaño declarado del animal 33
Figura 12: Muestra de imágenes disponibles. 34
Figura 13: Ejemplos de imágenes seleccionadas desde la vista posterior, empleadas como referencia morfológica para la estimación del peso 37
Figura 14: Diagrama desarrollo de Modelos Tradicionales. 39
Figura 15: Diagrama del MLP 43
Figura 16: Diagrama de Validación Cruzada 45
Figura 17: Diagrama de red multimodal o hibrida 47
Figura 18: Evolución del Loss (MSE) y Evolución del MAE. 49
Figura 19: Diagrama de flujo modelo multimodal en serie CNN + RF 50
Figura 20: Gráfico de Valores Reales Vs Predicciones 58
Figura 21: Histograma de Residuales 59
Figura 22: Curva Loss del Fine-Tuning 59
Figura 23: Interfaz Inicial 64
Figura 24: Resultado Obtenido 65
Figura 25: Tamaño que ocupa aplicación en dispositivo Android 67

LISTA DE TABLAS

Tabla 1: Principales estudios de predicción de datos usando AI y Ciencia de Datos 26
Tabla 2: Muestra de la tabla de datos tabulados 28
Tabla 3: Análisis descriptivos variables numéricas. 28
Tabla 4: Descripción de variables categóricas. 30
Tabla 5: Valores explorados y resultados de la búsqueda de parámetros para modelo SVR 41
Tabla 6: Valores explorados y resultados de la búsqueda de parámetros para modelo RF 41
Tabla 7: Valores explorados y resultados de la búsqueda de parámetros para modelo XGBOOST REGRESSOR 42
Tabla 8: Valores explorados y mejores resultados de hiperparámetros. 46
Tabla 9: Resultados Modelos Tradicionales 55
Tabla 10: Resultados Modelos Multimodales 55
Tabla 11: Desempeño y Tamaño de Aplicación en Dispositivo Android 67

LISTA DE ECUACIONES

Ecuación 1: Modelo de Aprendizaje Supervisado. 13
Ecuación 2: Regresión Lineal Múltiple. 14
Ecuación 3: Error Absoluto Media. 22
Ecuación 4: Raíz del Error Cuadrático Medio. 22
Ecuación 5: Error Cuadrático Medio. 22
Ecuación 6: Coeficiente de Determinación. 23

LISTA DE ANEXOS

Anexos 1: Fuente Datos. *73*
Anexos 2: Repositorio Archivos Modelos ML y CNN *73*
Anexos 3: Repositorio Archivos Modelos ML y CNN para Android. *73*
Anexos 4: Estimaciones de la Aplicación Android *75*

INTRODUCCIÓN

La industria ganadera enfrenta desafíos constantes para optimizar la productividad y garantizar transacciones comerciales justas. Una de las métricas fundamentales en este proceso es el peso vivo del animal; Sin embargo, en contextos de pequeños y medianos productores el acceso a básculas industriales es limitado debido a su alto costo, mientras que métodos manuales como la cinta bovinométrica pueden resultar riesgosos para el propietario o manipulador del animal. Tradicionalmente, la negociación se ha basado en la estimación visual subjetiva, lo que genera inequidades en el mercado.

En respuesta a esta problemática este proyecto desarrolló un sistema tecnológicamente accesible para la predicción del peso del ganado bovino utilizando técnicas de visión artificial y aprendizaje automático (*Machine Learning).* El enfoque se centró en aprovechar la capacidad de procesamiento de los dispositivos móviles modelos para ofrecer una herramienta de bajo costo y fácil uso en el campo.

Para lograr este objetivo, se procesó y analizó un conjunto de datos multimodal compuesto por registros tabulares e imágenes de 513 especímenes de algunas razas. Se implementaron y evaluaron diversos algoritmos de regresión, desde modelos clásicos como Support Vector Regression (SVR) hasta ensambles más robustos como Random Forest y XGBoost, así como arquitecturas de aprendizaje profundo (Deep Learning) utilizando redes neuronales convolucionales (CNN) pre entrenadas como ResNet50 y VGG16.

Los resultados obtenidos permitieron construir una arquitectura en serie, capaz de extraer características visuales del animal y procesarlas para estimar su peso. Finalmente, este modelo fue desplegado en un prototipo de aplicaciones móvil Android que opera mediante Edge Computing (TensorFlow Lite), eliminando la dependencia de conexión a internet y validando la aplicabilidad de la ciencia de datos en entornos agropecuarios reales.

1. DEFINICIÓN DEL PROBLEMA

1.1. Planteamiento del problema

La ganadería de vacuno es una actividad ampliamente desarrollada alrededor del planeta ya que es una actividad económica primaria que representa cerca del 40% de producción agrícola global, proporcionado el 34% de la ingesta de proteína y el 18% en energía a nivel mundial. Actualmente hay alrededor de 1.300 millones de personas que se dedican al cuidado y cría de ganado bovino, de los cuales se destacan 600 millones en condiciones de pobreza y vulnerabilidad. [1]

En la actualidad, el comercio del ganado bovino aún se realiza basado en la estimación de peso, los métodos más utilizados son la báscula y la cinta de peso, sin embargo, estos tienen sus desventajas, especialmente para los criadores pobres de zonas vulnerables y apartadas, pues son dispositivos costosos de implementar como el de la báscula, y en el caso de la cinta se torna impreciso porque depende de la raza de la res y es necesario que el animal sea dócil ya que se requiere medir el contorno del pecho de este. Es así, que el pequeño ganadero ha optado tradicionalmente a estimar el peso de la res por observación visual o estimación visual subjetiva.

El proceso de negociación del valor de un bovino mediante estimación visual subjetiva se realiza típicamente en el sitio de cría (potrero o corral), debido a la falta de infraestructura de pesaje. Este método se fundamenta en la observación directa y cercana del animal, donde la principal fuente de información es la apreciación empírica de ambas partes. El comprador y el vendedor intentan llegar a un acuerdo sobre el peso vivo del animal concentrándose en su conformación corporal, el estado de engrasamiento y la complexión general. Además de la estructura volumétrica, se consideran los rasgos fenotípicos, como la musculatura del tercio posterior y las características raciales visibles en la cabeza y el morro, que son indicadores de rendimiento cárnico en ciertas estirpes. Este método carece de estandarización y está sujeto a sesgos.

Este tipo de negociación generalmente termina siendo injusta para alguna de las partes y en mayor porcentaje de casos para el vendedor, ya que el comprador por experiencia de su actividad recurrente, acumula más información y puede tener una estimación mejor acertada pudiendo tomar ventaja y establecer el peso del animal a su favor, además de esto, en el evento de observación pueden existir afectaciones de apreciación relacionadas al lugar donde se realiza, no es lo mismo

dimensionar un objeto en un espacio abierto con pobres puntos de referencia a hacerlo en un espacio reducido y cerca del animal.

Es por lo anterior y aprovechando el auge de las herramientas tecnologías como la ciencia de datos se tornó interesante desarrollar una herramienta de clasificación y regresión que le permite a los criadores y comerciantes de ganado vacuno predecir el peso de estos mediante las técnicas de reconocimiento o detección de imágenes que permitan modelar la relación entre las características extraídas y el peso real del animal.

1.2. Formulación del problema

¿Cómo desarrollar un sistema para la predicción de peso de ganado bovino a través de visión artificial y técnicas de aprendizaje automático?

Sistematización:

¿Cómo preparar el data set de imágenes e información de ganado bovino, realizando el etiquetado de áreas del cuerpo y peso para cada individuo?

¿Cómo entrenar diferentes modelos de regresión, que permita identificar la relación entre las características extraídas de las imágenes y el peso real del animal?

¿Cómo evaluar los rendimientos de los diferentes modelos de regresión mediante métricas correspondientes?

¿Cómo desarrollar un prototipo de aplicación con una interfaz sencilla que permita cargar imágenes y evaluar su utilidad en el contexto definido?

El proyecto se centró en la creación de un sistema de estimación de peso de vacunos mediante un flujo de datos estructurado. Inicialmente, se procesó una base de datos de imágenes para extraer características visuales con un modelo de detección y etiquetado. Estas características se fusionaron posteriormente con métricas morfométricas. Finalmente, se procedió a la implementación y validación de algoritmos de regresión de aprendizaje automático para la estimación del peso, culminando con el despliegue del modelo resultante en una aplicación de campo funcional.

2. OBJETIVOS

2.1. Objetivo General

Desarrollar un sistema para la predicción de peso de ganado bovino a través de visión artificial y técnicas de aprendizaje automático.

2.2. Objetivos Específicos

- Preparar el data set de imágenes e información de ganado bovino, realizando el etiquetado de áreas del cuerpo y peso para cada individuo.
- Entrenar diferentes modelos de regresión, que permita identificar la relación entre las características extraídas de las imágenes y el peso real del animal.
- Evaluar los rendimientos de los diferentes modelos de regresión mediante métricas correspondientes.
- Desarrollar un prototipo de aplicación con una interfaz sencilla que permita cargar imágenes y evaluar su utilidad en el contexto definido.

3. MARCO TEÓRICO Y ANTECEDENTES

3.1. Marco Teórico

Para cumplir con los objetivos planteados, fue esencial analizar las metodologías tradicionales de estimación de peso en ganado bovino como el uso de básculas, cintas bovinométricas y métodos basados en fórmulas como los de Quetelet y Crevat, con el fin de identificar sus limitaciones y proponer mejoras mediante soluciones tecnológicas. Como parte de este análisis, se estudiaron técnicas avanzadas de machine learning, en particular los modelos tradicionales de regresión y redes neuronales convolucionales, que permiten estimar el peso a partir de imágenes o datos morfométricos con mayor precisión y escalabilidad. Además, se explicó un método para implementar estos modelos en dispositivos móviles, facilitando su uso en entornos agropecuarios con acceso limitado a infraestructura especializada. Todo el desarrollo se acompañó de una revisión de las métricas de evaluación del rendimiento y de la metodología estándar aplicable a proyectos de ciencia de datos, asegurando así un enfoque sistemático y reproducible.

3.1.1. Métodos de Estimación de Peso Vivo Bovino

Para estimar el peso del ganado bovino, existen tanto métodos tradicionales como enfoques tecnológicos modernos. Entre los métodos convencionales, la balanza o báscula sigue siendo la opción más precisa y confiable, aunque implica una inversión inicial alta. Como alternativa más accesible, se utiliza la cinta *bovinométrica*, que permite estimar el peso a partir del perímetro torácico con buena confiabilidad, siempre que se considere la raza del animal.

Además, existen métodos basados en fórmulas, como el de Quetelet, que combina el perímetro torácico y la longitud del cuerpo, aplicando distintas constantes según el sexo del bovino. Por otro lado, el método de Crevat incorpora también el perímetro abdominal, lo que ofrece una mayor precisión. Sin embargo, ambos métodos pueden ser menos fiables que la cinta bovinométrica y, al igual que esta, requieren esfuerzo físico para tomar las medidas al animal.

3.1.2. Fundamentos de Aprendizaje Automático (Machine Learning)

El aprendizaje automático (ML) es una rama de la inteligencia artificial (IA) y de la informática la cual se centra en el uso de los datos y los algoritmos, su aplicación práctica es la de imitar el modo como los humanos aprenden y mejorar gradualmente su precisión [7]. Autores como [8] [9] lo

definen como un conjunto de métodos que pueden detectar automáticamente patrones en los datos y tomar decisiones basadas en estos sin ser programados explícitamente para cada tarea.

El aprendizaje automático se basa en tres elementos esenciales: *Datos ("Experience"):* Son las observaciones o registros históricos que contienen la información relevante para alimentar el modelo de aprendizaje. *Modelo / Algoritmo ("Task"*): El conjunto de reglas matemáticas para aprender patrones o relaciones. *Medida de Desempeño ("Performance"):* Son la forma de evaluar qué tan bien aprende o predice el modelo. Estos componentes trabajan en conjunto para ajustar automáticamente los parámetros internos y mejorar su rendimiento con el tiempo [1].

El aprendizaje automático (ML) es un componente fundamental de la ciencia de datos. Mediante el uso de métodos estadísticos, permite entrenar algoritmos capaces de generar resultados como clasificaciones, predicciones y el descubrimiento de patrones o ideas clave en proyectos de minería de datos. Estos nuevos conocimientos respaldan la toma de decisiones en las organizaciones, lo que impacta positivamente en la generación de valor, la mejora de los indicadores clave de rendimiento y, en general, en el crecimiento de las compañías.

Aprendizaje Supervisado

Es el tipo de aprendizaje donde los datos son etiquetados y por ende la respuesta correcta ya se conoce, algunos de los ejemplos son la regresión lineal. En el aprendizaje predictivo o supervisado, tiene el objetivo de aprender una correspondencia entre las entradas x y las salidas y, dado un conjunto de pares de entrada-salida.

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$$

Ecuación 1: Modelo de Aprendizaje Supervisado.

Donde D se denomina conjunto de entrenamiento y N es el número de ejemplos de entrenamiento. Con el conjunto de datos etiquetados se entrena el algoritmo para clasificar los datos o predecir resultados con precisión, dado que a medida que los datos se introducen en el modelo este ajusta sus ponderaciones hasta lograr un nivel de ajuste adecuado. [9]

Etapa del Desarrollo de un Modelo

i. Recolección y limpieza de los datos, donde se recolectan datos relevantes y se preprocesan.
ii. Selección de características: se identifican los atributos más útiles o seleccionan las variables que aportan más significancia.

iii. Elección del algoritmo, se escoge el modelo con base en la tarea y el tipo de datos.
iv. Entrenamiento donde se ajustan los parámetros del modelo a los datos.
v. Evaluación, se prueba el rendimiento con datos nuevos en esta etapa se evalúa que tan generalizado está el modelo.
vi. Despliegue y monitoreo: Se implementa el modelo en una entorno y se vigila su desempeño continuo [2]

3.1.3. Modelos de Regresión Continua

La predicción de variables continuas es una tarea central en estadística y aprendizaje automático supervisado. Se basa en construir modelos que puedan estimar el valor de una variable dependiente $\boldsymbol{y}$ en función de una o más variables independientes $\boldsymbol{x_1}$, $\boldsymbol{x_2}$ … $\boldsymbol{x_n}$. Existen diversos enfoques para abordar este problema, desde métodos clásicos como la regresión lineal hasta técnicas más complejas como las redes neuronales y los métodos de ensamble.

Regresión Lineal Múltiple

Cuando se incluyen múltiples variables predictores, se utilizan la regresión lineal múltiple:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

Ecuación 2: Regresión Lineal Múltiple.

Este modelo es útil cuando se cree que varias características influyen simultáneamente en el valor de la variable objetivo. Sin embargo, es sensible a problemas como multicolinealidad y sobreajuste si no se aplica técnicas de regularización [3].

Algoritmos clásicos aplicados a la regresión continua.

A continuación, se describen algunas de las técnicas más usadas en el aprendizaje automático.

Máquinas de Vector de Soporte (Support Vector Machine SVM)

Es un algoritmo de aprendizaje automático supervisado que clasifica los datos al encontrar una línea o hiperplano optimo que separe las clases con la máxima distancia posible respecto a los puntos más cercanos llamados vectores de soporte. Esta característica le permite lograr una separación robusta y con buen poder de generalización en un espacio N-dimensional [4] [5].

SVM funciona asignando datos a un espacio de características de alta dimensión para que los puntos de datos se puedan clasificar, incluso cuando los datos no se pueden separar linealmente [10].

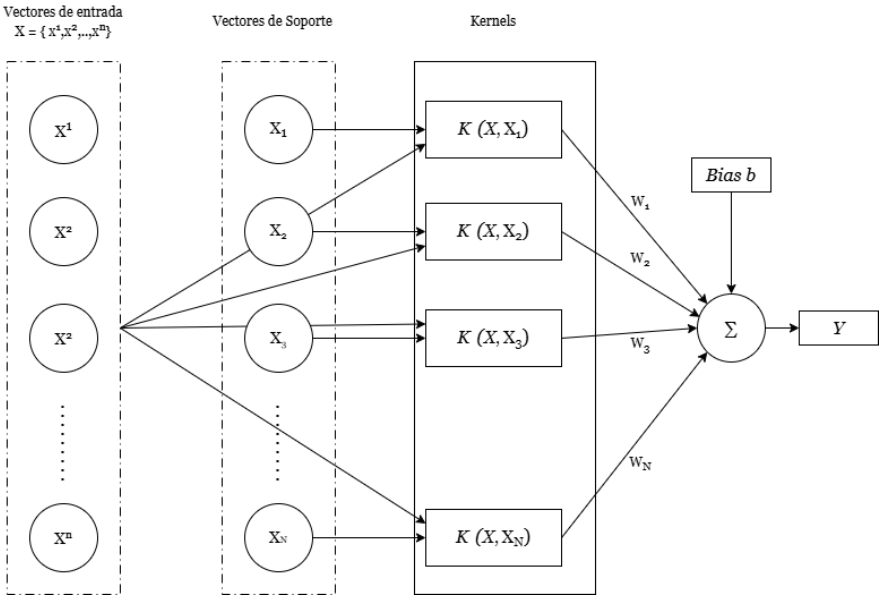


Figura 1: Arquitectura de una SVM de regresión

En el ámbito de la regresión, el modelo se adapta mediante Support Vector Regression (SVR), que utiliza el mismo principio de márgenes, pero enfocado en ajustar una función que prediga valores continuos dentro de un rango de tolerancia [6]. En lugar de clasificar datos, intenta encontrar una función que se mantenga dentro de un margen de tolerancia ε respecto a los datos reales, penalizando errores fuera del margen [7]. La figura 1 muestra la arquitectura del SVM para el caso de regresión.

Entre sus principales ventajas se destaca su eficacia en data sets pequeños y su capacidad de generalizar adecuadamente en problemas complejos con dimensiones elevadas, siempre que se elija un *kernel* apropiado. [8]

Bosque Aleatorio (Random Forest)

El Random Forest es un método de *ensemble learning* basado en la construcción de múltiples árboles de decisión, cuya salida se combina mediante votación (para clasificación) o promedios (para regresión como lo muestra la figura 2) [9]. Este enfoque mejora la estabilidad y precisión del modelo respecto a un único árbol.

En tareas de regresión, el algoritmo predice valores continuos a partir del promedio de las predicciones generadas por cada árbol, lo que reduce la varianza y mejora la capacidad de generalización [9], [10].

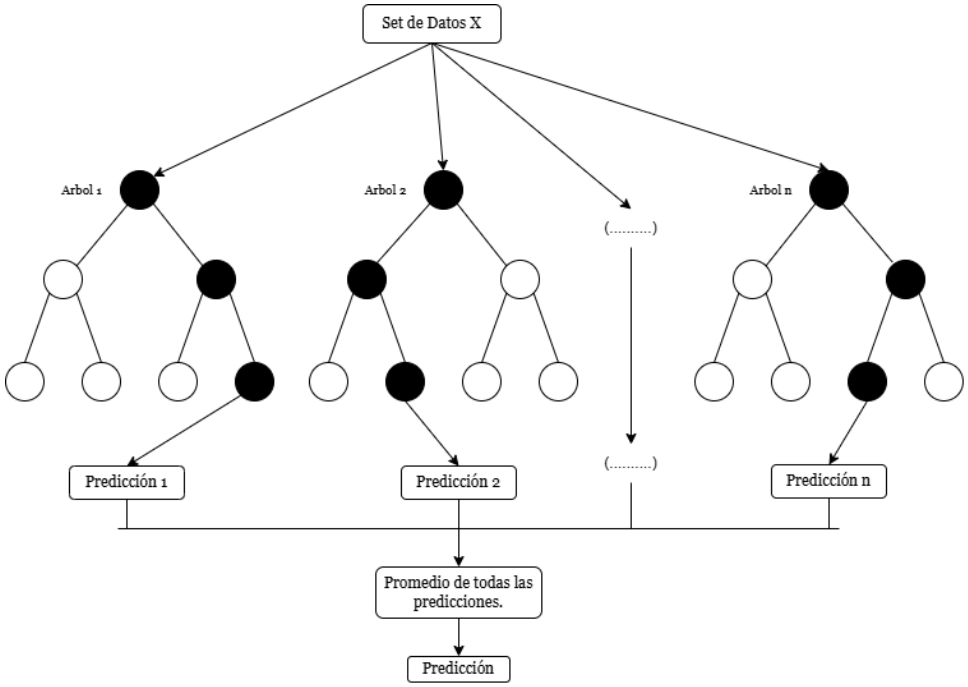


Figura 2: Arquitectura de Random Forest

Entre sus ventajas destacan la resistencia al sobreajuste, ya que combina múltiples modelos débiles, y su habilidad para capturar relaciones no lineales en los datos [9], [11].

Extreme Gradient Boosting (XGBoost)

El Extreme Gradient Boosting (XGBoost) es una implementación optimizada del algoritmo de gradient boosting, diseñada para ser altamente eficiente, flexible y escalable [12]. Su principio se basa en construir modelos aditivos de árboles de decisión de manera secuencial, donde cada nuevo árbol corrige los errores residuales de los anteriores [13].

En tareas de regresión, XGBoost ajusta un conjunto de árboles de decisión para minimizar una función de pérdida, utilizando técnicas de regularización que previenen el sobreajuste y mejoran la capacidad de generalización [12], [14] .

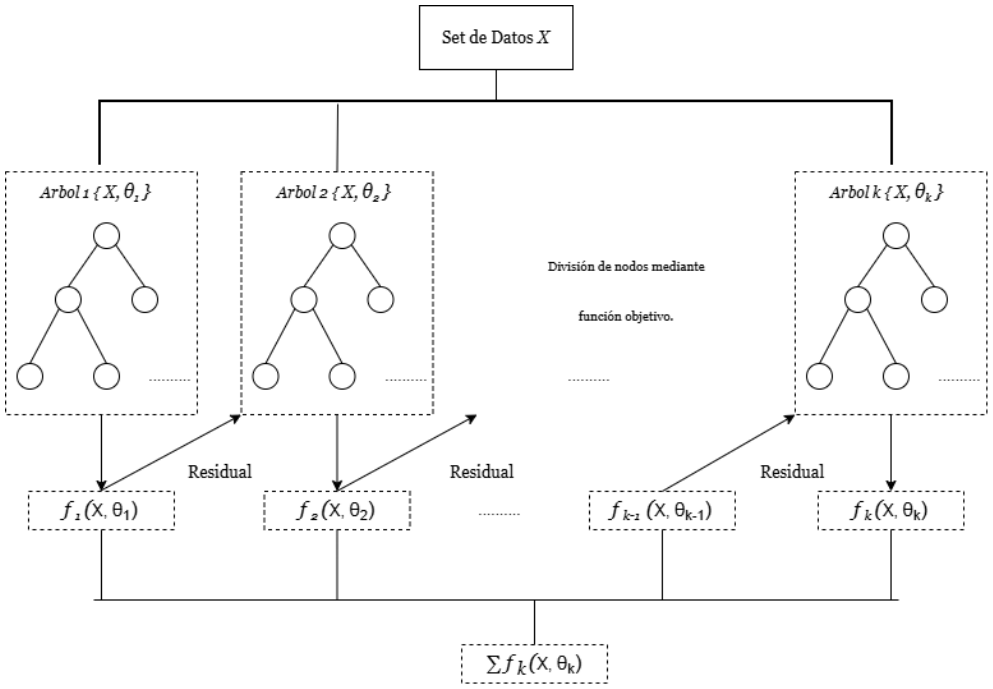


Figura 3: Arquitectura de XGBoost Regresor.

Entre sus ventajas se destacan su alto rendimiento computacional, debido a la optimización del uso de memoria y paralelización, así como su capacidad de manejar datos no lineales y con valores faltantes de manera efectiva [12], [13].

3.1.4. Redes Neuronales Artificiales

Las redes neuronales son un tipo de modelo matemático inspirado en el funcionamiento del cerebro humano, diseñadas para reconocer patrones y aprender de los datos. [2]

Perceptron Multicapa (MLP)

El perceptron Multicapa (MLP) es un tipo de red neuronal artificial compuesta por tres elementos principales: una capa de entrada, una o varias de capas ocultas, y una capa de salida [15] (representada en la figura 1). Cada neurona aplica una transformación lineal a la entrada, seguida de una función de activación (como sigmoid, thang o ReLU) que introduce no linealidad y permite representar relaciones complejas [1].

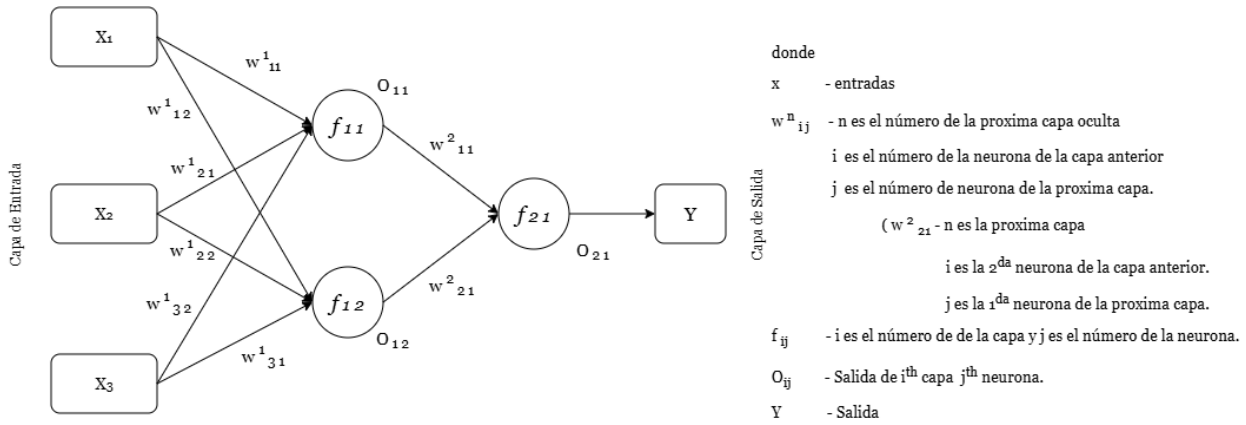


Figura 4: Red neuronal perceptrón multicapa

En problemas de regresión, los MLP son especialmente útiles para modelar relaciones no lineales entre las variables de entrada y la variable objetivo, lo que los convierte en una herramienta flexible y poderosa [1] [16].

Red Neuronal Convolucional (CNN)

Una Red Neuronal Convolucional (CNN, por sus siglas en inglés) es un tipo de modelo de aprendizaje profundo diseñado para procesar datos estructurados en forma de rejilla, como imágenes. Se caracteriza por el uso de capas convolucionales que aplican filtros para extraer características espaciales y jerárquicas de los datos de entrada, como bordes, texturas, formas y objetos completos, lo que la hace especialmente adecuada para tareas de visión por computadora, incluyendo la detección de imágenes.

Las CNN's se componen típicamente de tres tipos principales de capas, las *capas convolucionales*, que detectan patrones locales al aplicar filtros que recorren la imagen, las *capas de agrupamiento* (Pooling), que reducen la dimensionalidad manteniendo las características esenciales y las *capas completamente conectadas*, que integran las características extraídas para realizar clasificaciones o predicciones.

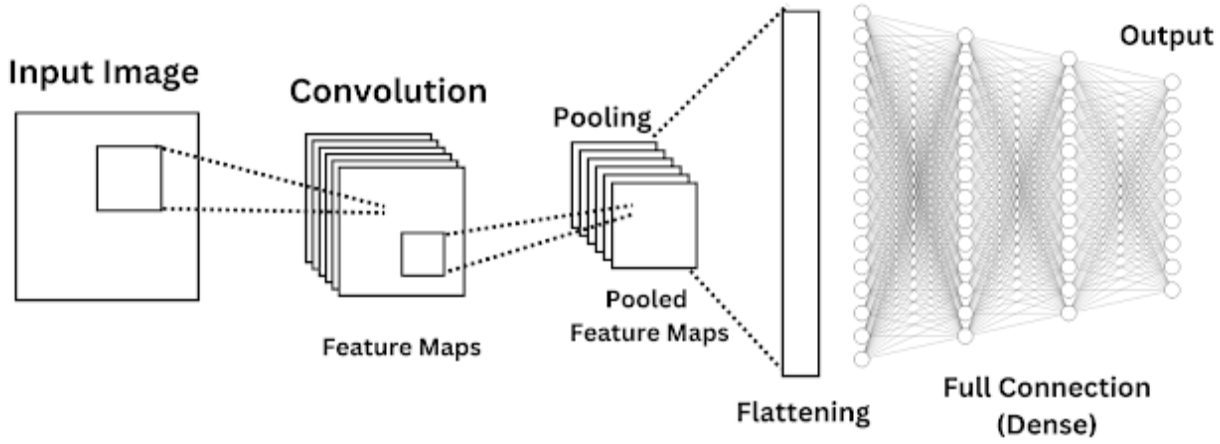


Figura 5: Arquitectura de una CNN.
Tomado de [17].

La implementación de estas redes en dispositivos móviles ha ampliado sustancialmente su campo de aplicación, permitiendo realizar inferencias en tiempo real directamente en el dispositivo, sin dependencia de conexión a servidores externos. Esta portabilidad las hace especialmente valiosas para aplicaciones en entornos remotos o con limitaciones de conectividad, como en explotaciones agropecuarias donde pueden utilizarse para estimar el peso del ganado mediante el simple análisis de imágenes capturadas con un smartphone.

3.1.5. Aprendizaje por transferencia

El Aprendizaje por transferencia (transfer learning) es una técnica de aprendizaje automático que consiste en reutilizar el conocimiento adquirido por un modelo entrenado en un gran conjunto de datos para resolver una nueva tarea con un conjunto de datos más pequeño [18]. En lugar de entrenar una red neuronal desde cero, se emplean arquitecturas pre entrenadas que ya han capturado representaciones útiles, las cuales se ajustan mediante un proceso de fine-tuning o bien se usan como extractores de características [19].

Entre sus beneficios principales se encuentra la mejora del rendimiento en escenarios con datos limitados, ya que permite aprovechar patrones previamente aprendidos y reducir el riesgo de sobreajuste. Esto resulta especialmente relevante en el contexto de este proyecto, donde el número de muestras disponibles es reducido, y el uso de redes pre entrenadas optimiza la precisión y eficiencia del modelo [18], [20].

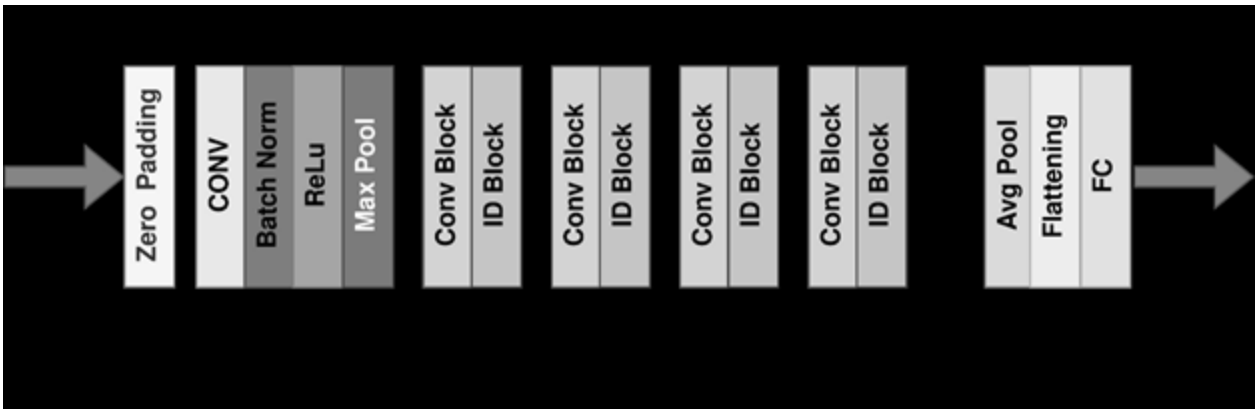


Figura 6: Arquitectura de la Resnet50
Tomado de **[21]**.

En el desarrollo del presente proyecto se utilizó la red pre entrenada Resnet50 y VGG16, La ResNet-50 es una arquitectura profunda de redes neuronales convolucionales basada en bloques residuales, cuyo objetivo es facilitar el entrenamiento de modelos muy profundos mediante el uso de conexiones de atajo (skip connections). ResNet-50 consta de 50 capas, organizadas en una secuencia de bloques bottleneck que reducen la dimensionalidad antes de aplicar convoluciones más costosas, lo que mejora la eficiencia computacional sin sacrificar capacidad de representación. [22]. Por otro lado la VGG16 cuenta con una arquitectura clásica (mostrada en la figura 7) de redes neuronales convolucionales desarrollada por el Visual Geometry Group de la Universidad de Oxford. Su diseño se caracteriza por el uso de **capas convolucionales profundas con filtros pequeños (3×3)** y capas de *pooling* que reducen progresivamente la dimensionalidad, permitiendo una representación jerárquica y detallada de las imágenes [23].

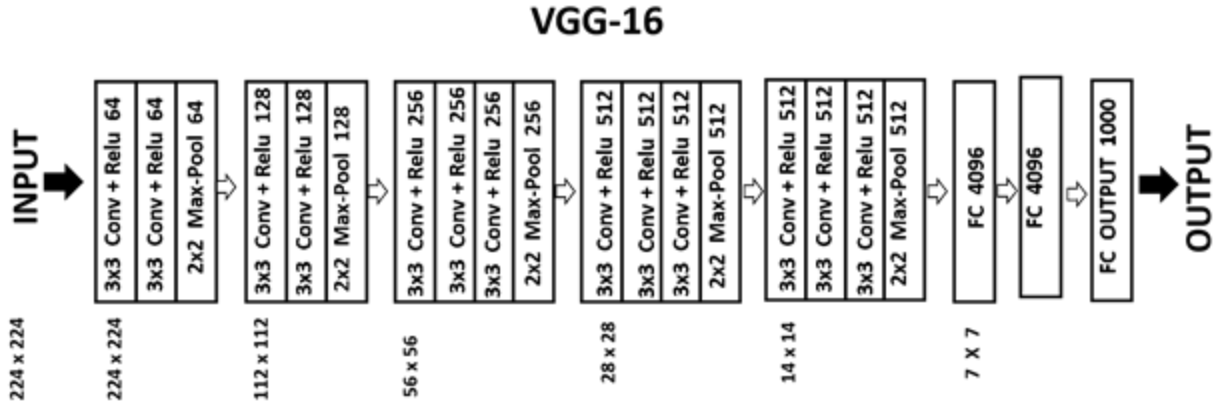


Figura 7: Arquitectura de la VGG16
Tomado de **[24]**.

Una de las aplicaciones más comunes de la VGG16 es como extractor de características dentro de esquemas de *transfer learning*, donde las representaciones aprendidas a partir de grandes bases de datos, como *ImageNet*, se reutilizan para resolver tareas específicas [23], [25].

3.1.6. Modelos Multimodal (Hibrido)

El concepto de aprendizaje multimodal (multimodal learning) se refiere a la integración de diferentes fuentes de información, como datos tabulares, imágenes, texto o audio, en un único modelo predictivo. Su objetivo es aprovechar las complementariedades entre modalidades para obtener representaciones más ricas y, en consecuencia, predicciones más precisas [26]. En el caso de imágenes y datos estructurados, las redes neuronales convolucionales (CNN) pueden encargarse de la extracción de características visuales, mientras que algoritmos de regresión o redes densas procesan las variables tabulares, combinando ambos espacios de representación en un modelo conjunto [27].

La justificación de este enfoque en el ámbito de la estimación del peso del ganado radica en que el peso no depende únicamente de la información visual del animal (como su silueta o proporciones), sino también de características adicionales como la edad, la raza, el historial de alimentación o el estado fisiológico. Al integrar ambos tipos de datos, el modelo es capaz de capturar patrones que serían invisibles si se considerara una sola modalidad [28].

3.1.7. Validación Cruzada

La validación cruzada es una técnica de evaluación y validación de modelos de aprendizaje automático que se utiliza para medir su capacidad de generalización, es decir, su rendimiento al enfrentarse a datos nuevos y no vistos durante el entrenamiento. El principio fundamental de la validación cruzada es dividir el conjunto de datos en múltiples subconjuntos (o folds) y entrenar y validar el modelo en distintas combinaciones de estos subconjuntos.

El proceso típico implica los siguientes pasos, Dividir los datos en k subconjuntos aproximadamente iguales (folds), entrenar el modelo en $k-1$ folds y validar su rendimiento en el fold restante, repetir este proceso k veces, alternando los folds utilizados para validación y entrenamiento finalmente promediar los resultados de las validaciones para obtener una métrica robusta del rendimiento del modelo.

Esta técnica ayuda a mitigar problemas como el sobreajuste, al evaluar el modelo en datos que no se usaron para su entrenamiento. Además, es especialmente útil cuando el conjunto de datos es pequeño, ya que maximiza el uso de los datos disponibles. [14] [15].

3.1.8. Métricas de Evaluación de Modelos

Las siguientes herramientas de evaluación del desempeño de modelos de *machine learning* están especialmente orientados a las tareas de regresión.

Error Absoluto Medio (MAE): Métrica que mide la magnitud promedio del error entre las predicciones y los valores reales, sin considerar el signo. Es fácil de interpretar y se presenta en las mismas unidades que la variable objetivo, siendo útil para evaluar la precisión general del modelo.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Ecuación 3: Error Absoluto Media.

Tomado de [16].

Raíz del Error Cuadrático Medio (RMSE): Penaliza errores grandes al elevar al cuadrado las diferencias antes de calcular la raíz cuadrada. Se utiliza ampliamente porque combina sensibilidad a errores grandes con interoperabilidad en las mismas unidades del variable objetivo.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Ecuación 4: Raíz del Error Cuadrático Medio.

Tomando de [17].

Error Cuadrático Medio (MSE): Calcula el promedio de los errores al cuadrado, lo que amplifica las diferencias grandes entre las predicciones y los valores reales. Es útil para optimización durante el entrenamiento de modelos, pero menos interpretable que el RMSE.

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

Ecuación 5: Error Cuadrático Medio.

Tomado de [18].

Coeficiente de Determinación (R^2): Mide qué porcentaje de la variabilidad de los datos es explicado por el modelo, con valores cercanos a 1 (uno) indicando un buen ajuste. Es una métrica estándar para comparar modelos en términos de su capacidad explicativa.

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

Ecuación 6: Coeficiente de Determinación.

Tomado de [19].

3.1.9. Modelos ML/CNN en Móvil

El **Edge Computing** es un paradigma de computación distribuida en el que el procesamiento y almacenamiento de datos se realiza lo más cerca posible de la fuente de generación (en el "borde" de la red), en lugar de enviarlos a un centro de datos centralizado. Esto permite reducir la latencia y aliviar la carga de la nube, aprovechando recursos locales para responder con rapidez. Los "edge devices" son los nodos o dispositivos que forman parte de esta arquitectura perimetral, ya que son capaces de generar datos y ejecutar procesos de cómputo local. **TensorFlow Lite** es una versión ligera de TensorFlow diseñada específicamente para ejecutar modelos de machine learning directamente en dispositivos móviles y embebidos, como teléfonos Android o dispositivos IoT. Su arquitectura está optimizada para reducir el tamaño del modelo y minimizar el uso de memoria, empleando un intérprete compacto que soporta aceleración por hardware (como la API Neural Networks de Android). Además, TensorFlow Lite permite tanto la inferencia ("run") como el entrenamiento en el dispositivo (on-device training), lo que facilita la personalización de modelos sin necesidad de depender únicamente de servidores remotos.

Android Studio es el entorno de desarrollo oficial (IDE) para crear aplicaciones para Android. Según estudios académicos, se utiliza para diseñar la interfaz de usuario, desarrollar la lógica de la app, compilar el código y desplegarlo en dispositivos reales o emulados, integrando bibliotecas y modelos de aprendizaje automático. Este entorno es especialmente útil para incorporar modelos de machine learning optimizados con TensorFlow Lite dentro de aplicaciones móviles, ya que permite trabajar con APIs de Java o C++ para cargar e invocar modelos TFLite.

3.2. Antecedentes

A continuación, se describen los principales trabajos de investigación que guardan relación con el objeto de estudio y presentan un marco de antecedentes en el uso de la ciencia de datos para la detección y estimación de valores.

Estudio	Aporte
MultiCamCows2024 - A Multi-view Image Dataset for AI-driven Holstein-Friesian Cattle Re-Identification on a Working Farm A**utores:** Phoenix Yua , Tilo Burghardta , Andrew W Dowseyb , Neill W Campbella	Este trabajo consistió en desarrollar un sistema para identificar vacas que caminan en un corral, en total 90 vacas, mediante la implementación de 3 cámaras se realizaron la implementación de la visión computacional llevando el modelo a reconocer patrones de la mancha en el lomo del animal que en este caso fue una raza especifica Holstein-Friesian, El objetivo general del sistema es ayudar a monitorear el comportamiento de los animales en el corral. Para nuestro proyecto los principales aportes que tiene este trabajo es que ayudan a entender cuando debemos evaluar la implementación de un sistema de identificación supervisado o no supervisado. En este caso, el supervisado va más alineador a nuestro desarrollo esto lo que nos conduce es a la dependencia de recursos fijos mientras que el no supervisado lo que permite es la introducción de nuevos individuos sin que se deba hacer el etiquetado manual para cada uno, lo que en el trabajo se le llama re identificación, lo cual, sin embargo no sería aplicable al presente proyecto, ya que los registros visuales de los animales deben ir acompañados de una variable más, la cual es el peso y por temas de entrenamiento del modelo se debe verificar manualmente. [23]
Cattle Body Detection Based on YOLOv5-EMA for Precision Livestock Farming **Autores:** Wangli Hao, Chao Ren, Meng Han , Li Zhang, Fuzhong Li * and Zhenyu Liu	Este objetivo principal de este proyecto fue la implementación de YOLOV5 incorporando el uso del módulo Efficient Multi-Scale Attention (EMA), el cual es la columna vertebral de los modelos de detección de la serie YOLO, esto con el fin de detectar y clasificar características

	importantes en el ganado bovino con mayor precisión tanto del cuerpo como de las extremidades en múltiples escenarios, enfocándose principalmente en la cabeza y patas, empleando un enfoque de fusión de múltiples características para capturar bordes, valores de escala de grises y relaciones de posición espacial de los animales. [24]. Este trabajo lo que nos ayuda es a entender cómo abordar y que herramientas utilizar específicamente para lograr detectar las partes del cuerpo de nuestro interés. Además de que puede ayudarnos a solucionar una de las principales preocupaciones y es que el modelo detecte en múltiples escenarios.
PREVISIÓN DEL PESO VIVO EN ANIMALES HOLSTEIN MEDIANTE BARIMETRÍA: ECUACIONES DE PREDICCIÓN SIMPLES Y MÚLTIPLES **Autores:** García Lara, I.1 , Vázquez Ferreño, M.A.2 , Fernández Calviño, E.3 , Vidal Galego, L.3 y García Lara, M.T de J.1	El propósito de este trabajo fue realizar un estudio a novillas de la raza Holstein, donde lo que se busco fue comprobar la correlación que existe entre el peso y las variables de la altura de la cruz (dimensión del suelo al espalda zona cruz), perímetro torácico, y edad del animal usando los modelos de regresión simple y múltiple lo que arrojo en orden de relevancia que el perímetro torácico tiene la mayor correlación seguido de la altura de la cruz y en mucha menor medida la edad. [25] Para propósito de nuestro proyecto, esto nos ayuda a aclarar y a definir que el perímetro torácico es la variable principal a identificar.
Image dataset for cattle biometric detection and analysis. **Autores:** Lili Bai, Zhe Zhang, Jie Song.	El objetivo de este informe fue describir principalmente como hacer la recolección de los datos, se nombran las herramientas que se usaron, el software que se usó para hacer las etiquetas de entrenamiento de las imágenes o datos que en este

	caso fue el Labellmg, como tomar las medidas que interesan a la hora de estimar el peso y consideraciones relevantes a la hora de entrenar el modelo de detección de objetos, finalmente ofrece algunas consideraciones de cómo se puede aprovechar los datos procesados en beneficio de los granjeros [26]. Para propósitos de este proyecto, este informe da un importante contexto de la metodología de adquirir los datos requeridos, en el casi que sean requeridos más de los disponibles, esto con el propósito de hacer más robusto el modelo.
Development of an automatic cow body condition scoring using body shape signature and Fourier descriptors **Autores:** A. Bercovich ,* Y. Edan ,* V. Alchanatis ,† U. Moallem ,‡ Y. Parmet ,* H. Honig ,‡ E. Maltz ,† A. Antler ,† and I. Halachmi †1	Este informe de investigación, describe la metodología para estimar la huella de la forma corporal de una vaca sobre la cabeza de cola, ya que para los veterinarios la forma de esta puede reflejar condiciones importantes de salud como por ejemplo estimar el estado de salud y energía de reserva del animal (gorda o flaca). El método implementado no se enfoca en la implementación de una herramienta de inteligencia artificial ya desarrollada, sino que procesa fotos de la cola de la vaca desde una vista superior, este perfil, con la aplicación de filtros de color y descripción de Fourier puede generar el contorno con dimensiones de la zona a interés [27]. Para este proyecto, el método podría ser considerado, ya que, si el objetivo es encontrar dimensiones sobre el bovino, una forma de hacerlo es mediante el método descrito por este trabajo.

Tabla 1: Principales estudios de predicción de datos usando AI y Ciencia de Datos

4. OBTENCIÓN Y PROCESAMIENTO DE LOS DATOS

4.1. Adquisición de los datos

Los datos utilizados fueron obtenidos del repositorio público de Github [29], declarado como de uso público con el fin de facilitar su reutilización en investigaciones posteriores. El conjunto contiene un total de 17,899 imágenes correspondientes a 513 registros. Cada registro se encuentra asociado a un conjunto de datos tabulados complementarios que incluyen 11 atributos, los cuales se describen en la siguiente sección.

4.2. Descripción general

El conjunto de datos está compuesto por 513 registros correspondientes a especímenes bovinos. Cada registro incluye variables descriptivas relacionadas con características físicas (peso, altura, edad, dentadura), atributos categóricos (raza, color, sexo, tamaño) y metadatos sobre el número de imágenes asociadas. En total se registran 14 atributos, de los cuales 8 son numéricos y 6 categóricos.

La tabla 2 muestra la estructura del conjunto de datos, donde la columna sku contiene identificadores únicos asociados a cada ejemplar bovino y sus imágenes correspondientes. Las columnas restantes registran características morfométricas y atributos relevantes de los animales, entre los cuales destaca el peso del animal, codificado en la columna como:

- *weight_in_kg* como variable objetivo para el modelo predictivo.
- *height_in_inch, age_in_year, teeth:* Variables morfometricas críticas para la estimación de peso.
- *sex, breed:* Variables de clasificación animal muy relevantes.

Nombre de la Variable	**Tipo de Dato**	**Subtipo**	**Descripción**
sku	string	Identificador	Identificador único del animal.
sex	categorica	Nominal	Sexo del animal.
color	categorica	Nominal	Color del pelaje del animal.
breed	categorica	Nominal	Raza del animal.
feed	categorica	Nominal	Tipo de alimentación o dieta.
age_in_year	numérica	Continua	Edad del animal en años.

teeth	numérica	Discreta	Numero de dientes del animal.
height_in_inch	numérica	Continua	Altura del animal en pulgadas
weight_in_kg	numérica	Continua	Peso del animal en kilogramos. **(Variable objetivo).**
price	numérica	Continua	Precio del animal en unidades monetarias.
size	categórica	Ordinal	Tamaño del animal (ej. Pequeño, Mediano, Grande).
Images_count	numérica	Discreta	Número de imágenes regulares asociadas.
yt_images_count	numérica	Discreta	Número de imágenes de un tipo específico.
total_images	numérica	Discreta	Total de imágenes asociadas al animal.

Tabla 2: Muestra de la tabla de datos tabulados

4.2.1. Exploración de Datos Tabulares

Análisis descriptivo univariante

En la tabla 3, se presenta la descripción de la distribución de cada variable numérica donde se pueden detallar las métricas estadísticas relevantes:

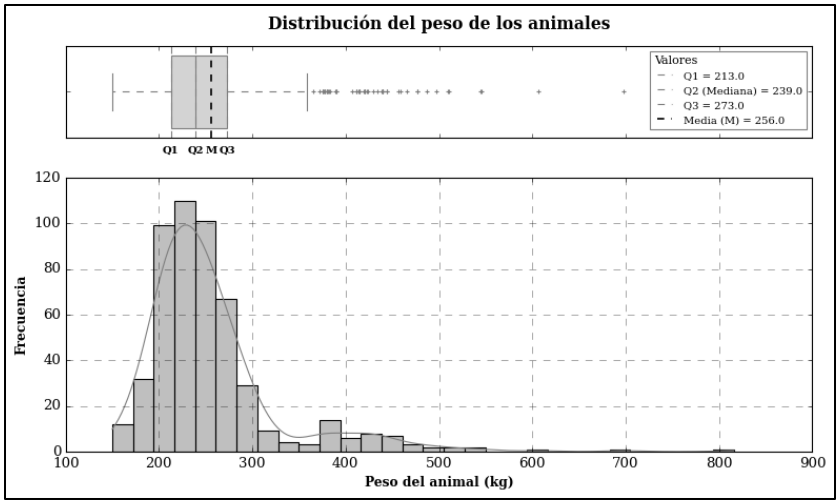
Variable	Conteo	Media	Desv. Est.	Mínimo	P25	Mediana (P50)	P75	Máximo
age_in_year	513	2.13	0.23	2.0	2.0	2.0	2.5	3.0
teeth	513	2.53	0.94	2.0	2.0	2.0	4.0	6.0
height_in_inch	513	46.67	3.24	38.0	44.3	46.6	48.4	58.2
weight_in_kg	513	256.03	74.30	150.0	213.0	239.0	273.0	816.0
price	513	106,861.07	33,684.31	16,000.0	91,000.0	101,000.0	116,000.0	335,000.0
images_count	513	4.00	0.00	4.0	4.0	4.0	4.0	4.0
yt_images_count	513	30.82	1.97	0.0	31.0	31.0	31.0	31.0
total_images	513	34.82	1.97	4.0	35.0	35.0	35.0	35.0

Tabla 3: Análisis descriptivos variables numéricas.

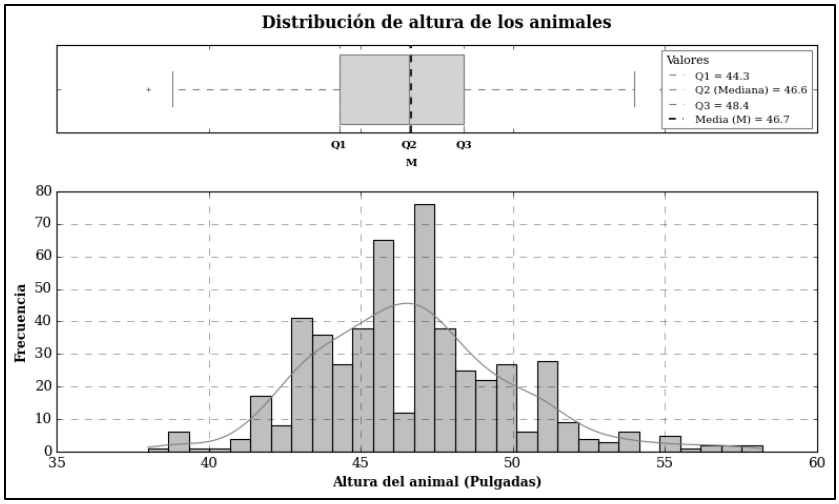
Interpretación de la tabla 3:

- Las variables *age_in_year* y *teeth* muestran poca variabilidad, indicando una población relativamente homogénea en edad y dentición.

- Altura (*height_in_inch*) y peso (*weight_in_kg*) presentan una distribución más amplia como se puede complementar con el grafico adicional de las figuras 8, lo que puede reflejar diferencias por raza, alimentación o crecimiento.



a)



b)

Figura 8: a) Distribución de frecuencia de la variable Altura (height_in_inch). b) Distribución de frecuencia variable peso (weight_in _kg).

- El peso y el precio exhiben valores máximos atípicos, lo que sugiere la existencia de ejemplares de mayor valor comercial.
- Las variables relacionadas con imágenes (*images_count*, *yt_images_count*, *total_images*) son casi constantes, salvo algunos registros con valores extremos, lo que indica homogeneidad en el material visual disponible.

Distribución de variables categóricas

Variable	Categoría	Frecuencia	Porcentaje (%)
sex	MALE_BULL	507	98.83
	FEMALE_HEIFER	6	1.17
color	RED	308	60.04
	NON_RED	205	39.96
breed	LOCAL	353	68.81
	SAHIWAL	59	11.50
	SINDHI	38	7.41
	HOSTINE_CROSS	32	6.24
	RED_CHITTAGONG	21	4.09
	PABNA_BREED	6	1.17
	BRAHMA	2	0.39
	MIR_KADIM	2	0.39
feed	['JUMBOO', 'LUCERNE', 'NAPIER', 'SILAGE', 'STRAW']	513	100.00
size	MEDIUM	320	62.38
	MINIMUM	147	28.65
	LARGE	37	7.21
	EXTRA_LARGE	9	1.75

Tabla 4: Descripción de variables categóricas.

Interpretación.

Sexo: El conjunto de datos está fuertemente sesgado hacia machos (98.83%), lo que puede reflejar una muestra orientada a animales de mayor valor comercial o productivo.

Color: Predomina el color rojo (60.04%), aunque existe una proporción considerable de ejemplares "non-red".

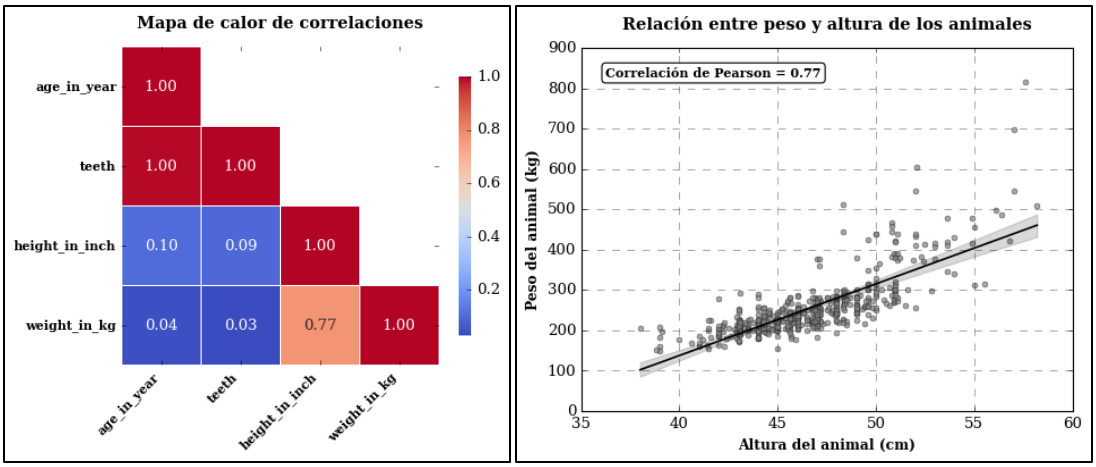
Raza: La raza LOCAL representa más de dos tercios de los registros, seguida de Sahiwal (11.5%) y Sindhi (7.4%), lo que indica un dominio de razas nativas.

Alimentación (feed): Todos los registros comparten el mismo patrón alimenticio, lo que sugiere que esta variable no aporta variabilidad significativa.

Tamaño: El tamaño medium es el más frecuente (62.38%), seguido de minimum (28.65%), reflejando una población con predominio de ejemplares medianos.

Análisis Bi-variado

La *figura 9* presenta un análisis multivariado orientado a la comprensión de las relaciones lineales entre las principales variables numéricas del conjunto de datos: edad (*age_in_year*), número de dientes (*teeth*), altura (*height_in_inch*) y peso (*weight_in_kg*).



a) *b)*

Figura 9: a) Mapa de color de correlación entre las variables numéricas. b) Relación entre la altura y peso del animal.

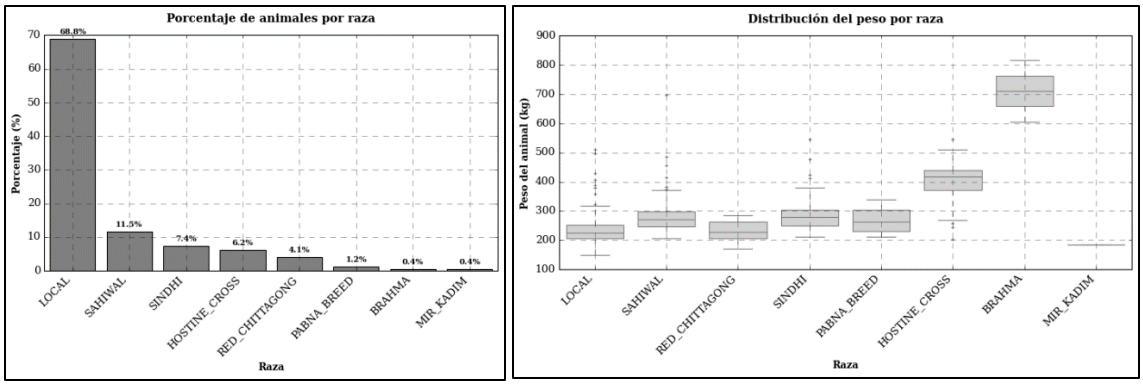
En el subgráfico (a) se muestra el mapa de calor de correlaciones de Pearson, que cuantifica la fuerza y dirección de las relaciones entre las variables. Se observa una correlación positiva moderadamente alta ($r = 0.77$) entre la altura y el peso, lo cual indica que a medida que aumenta la altura del animal, el peso tiende a incrementarse de manera proporcional. En contraste, las variables edad y número de dientes presentan correlaciones muy bajas con el peso ($r < 0.1$), lo que sugiere una influencia marginal de estas características sobre la masa corporal dentro del rango etario analizado.

El subgráfico (b) representa la relación directa entre la altura y el peso de los ejemplares mediante un diagrama de dispersión con ajuste lineal (regresión simple). Se aprecia una tendencia ascendente coherente con el coeficiente de correlación, evidenciando una dependencia lineal positiva significativa. La dispersión relativamente homogénea de los puntos alrededor de la línea de tendencia sugiere un comportamiento estable entre ambas variables, sin presencia marcada de heterocedasticidad.

Este análisis confirma que la altura (*height_in_inch*) constituye una de las variables predictoras más relevantes del peso (*weight_in_kg*), validando su uso como atributo clave en modelos posteriores de estimación o predicción del peso corporal.

Variables Categóricas.

La *figura 10* resume el comportamiento del peso de los animales en función de dos variables categóricas: raza **(***breed***)** y tamaño **(***size***)**, con el fin de identificar diferencias morfológicas relevantes en el conjunto de datos.



a) *b)*

Figura 10: a) Porcentaje de animales por razas. b) Distribución de por raza.

En el subgráfico (a) se muestra la distribución porcentual de animales por raza, observándose un predominio de la raza Local, que representa aproximadamente 68.8 % del total de registros. Le siguen las razas Sahiwal (11.5 %), Sindhi (7.4 %) y Hostine Cross (6.2 %), mientras que las razas Brahma, Pabna Breed y Mir Kadim tienen una participación marginal (< 2 %). Esta composición indica una base muestral dominada por razas locales, lo que podría reflejar la estructura productiva regional.

El subgráfico (b) presenta la distribución del peso por raza, evidenciando variaciones significativas entre grupos. Las razas Brahma y Hostine Cross exhiben los mayores valores promedio de peso, alcanzando hasta 700–800 kg, mientras que las razas Local y Red Chittagong muestran promedios cercanos a los 250–300 kg. La dispersión observada en algunas categorías sugiere diferencias intrarraza posiblemente asociadas a factores de alimentación o edad.

En general, se identifica una tendencia ascendente del peso en razas de mayor tamaño corporal, lo que es coherente con su clasificación zootécnica.

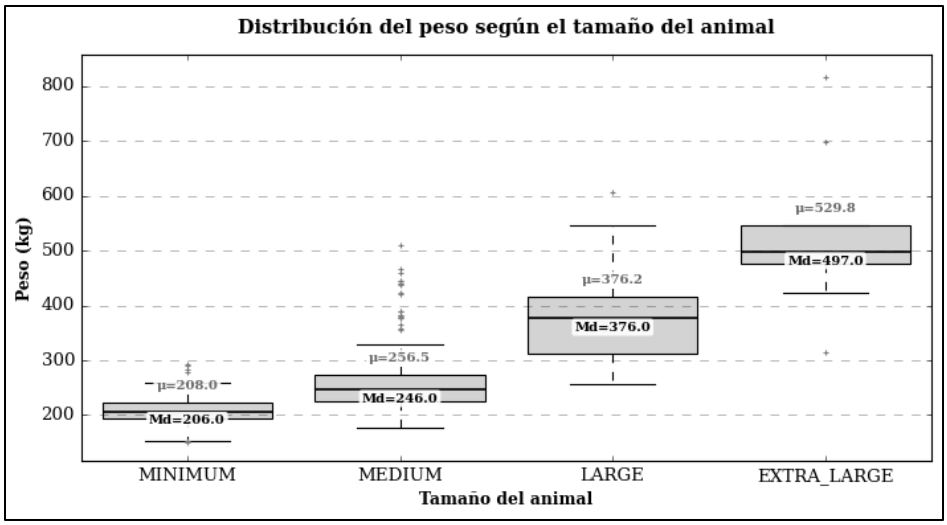


Figura 11: Distribución del peso según el tamaño declarado del animal.

Por su parte, la *figura 11* compara la distribución del peso según el tamaño declarado del animal (Minimum, Medium, Large, Extra Large). La gráfica confirma una relación directa entre tamaño y peso, con incrementos graduales tanto en la media (μ) como en la mediana (Md). Los animales clasificados como Extra Large presentan un peso promedio de 529.8 kg y mediana de 497 kg, frente a los Minimum, con promedios cercanos a 208 kg. La progresión lineal de ambas medidas estadísticas valida la coherencia interna del conjunto de datos y refuerza la consistencia entre las variables morfológicas y la clasificación categórica de tamaño.

Conclusión Técnica

El análisis categórico evidencia que el peso de los animales está fuertemente condicionado por la raza y el tamaño, siendo estas dos variables categóricas predictoras clave para modelos de estimación de peso. Las razas de mayor porte (como *Brahma* y *Hostine Cross*) y las categorías de tamaño *Large* y *Extra Large* concentran los mayores valores, mostrando coherencia entre las variables cualitativas y las métricas cuantitativas del conjunto de datos.

4.2.2. Imágenes del Set de Datos

El conjunto de imágenes asociado al data set corresponde a la toma fotográfica registrada para cada ejemplar identificado en la columna sku, la cual actúa como clave única que vincula la información

tabular con los archivos visuales. En total, el data set cuenta con 17,899 imágenes distribuidas en 513 registros, lo que representa un promedio aproximado de 35 imágenes por animal.

Cada grupo de imágenes muestra un mismo ejemplar capturado desde distintos ángulos y posiciones (frontal, lateral, posterior y oblicuo), como se observa en la *figura 12*. Este enfoque busca documentar de manera completa la morfología del animal, permitiendo observar detalles como la estructura corporal, la alzada y el volumen general, que son relevantes para estudios de estimación de peso corporal y análisis morfo métrico.

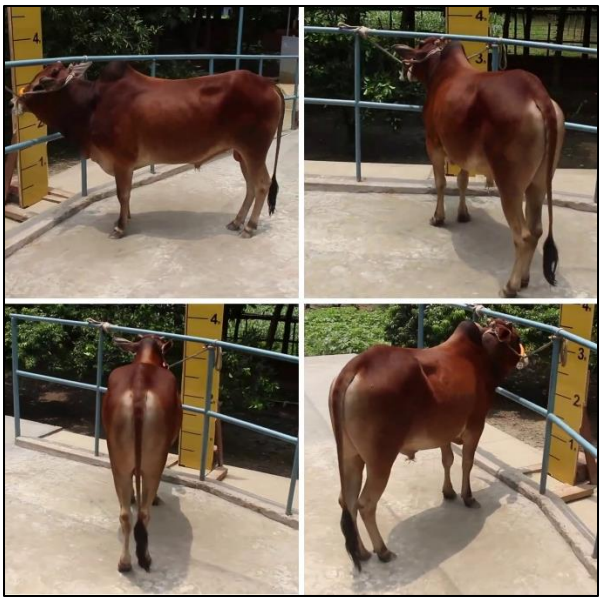


Figura 12: Muestra de imágenes disponibles.

Las fotografías se encuentran en formato .JPG y fueron tomadas en entornos controlados, con iluminación natural y fondos relativamente uniformes, lo cual facilita su posterior procesamiento. Este conjunto visual complementa la información tabular del conjunto de datos y será utilizado para extraer características de forma y proporción en futuras etapas del trabajo, orientadas al desarrollo de modelos predictivos basados en visión computacional

4.3. Procesamiento Inicial de Datos

El procesamiento inicial tuvo como propósito garantizar la consistencia y calidad del conjunto de datos antes del análisis exploratorio y la etapa de modelado. Para ello se realizaron diferentes tareas de verificación, limpieza y estructuración de los datos.

En primer lugar, se revisó la integridad de los registros mediante un control de valores nulos, duplicados y formatos de datos. Se identificaron 2 registros incompletos, presentes en los datos tabulares pero no encontrados en las imágenes, por lo que fue necesario eliminar filas por ausencia de información. Además, se detectó la presencia de columnas con valores constantes, como *images_count*, que mantuvieron el mismo valor en todos los registros (4 imágenes base por animal). Aunque no fueron eliminadas, estas variables fueron marcadas como atributos sin variabilidad analítica y se descartaron para el análisis.

4.3.1. Transformación de los Datos.

El procesamiento inicial de los datos se realizó con el propósito de garantizar la calidad y homogeneidad del conjunto de información previo a su modelado.
El procedimiento implementó una secuencia de transformaciones orientadas a la limpieza, codificación y estandarización de las variables del conjunto de datos bovino.

Como etapa inicial del pre procesamiento, se verificó la tipología de las variables del conjunto de datos, asegurando que cada una correspondiera correctamente a su naturaleza numérica o categórica. Posteriormente, se realizó la conversión de los tipos de datos cuando fue necesario, con el fin de garantizar su adecuada interpretación por parte de los modelos de aprendizaje automático.

Tras esta verificación, se constató que el conjunto de datos no presentaba valores faltantes, por lo que no fue necesario aplicar técnicas de imputación.

En relación con las variables numéricas (*edad, altura, peso y precio*), se aplicó el método *StandardScaler*, llevando a cabo un proceso de estandarización (media igual a *cero* y desviación *estándar unitaria*). Este procedimiento permitió homogeneizar las escalas de las variables, evitando que aquellas con mayor magnitud influyeran desproporcionadamente en el proceso de aprendizaje y contribuyendo a una mayor estabilidad y eficiencia en algoritmos sensibles a la escala, como los modelos de regresión y las redes neuronales.

Por su parte, las variables categóricas (sexo, color, raza y tamaño) fueron transformadas mediante el método *One-Hot Encoding*, lo que permitió representarlas en formato numérico sin introducir relaciones ordinales artificiales ni pérdida de información semántica.

Finalmente, el conjunto de datos procesado se dividió en subconjuntos de entrenamiento y prueba, los cuales se almacenaron en formato .csv para su utilización en la etapa de modelado. El resultado final es un conjunto de datos limpio, balanceado y preparado para análisis predictivos, asegurando la consistencia de las variables tanto categóricas como numéricas.

4.3.2. División de Datos

Para garantizar una evaluación objetiva del rendimiento del modelo y evitar el sobreajuste, el conjunto de datos fue dividido en dos subconjuntos principales: entrenamiento (train) y prueba (test).

Esta división permite que el modelo aprenda patrones a partir de una porción de los datos y, posteriormente, se evalúe con información que no ha sido utilizada durante el aprendizaje, asegurando así su capacidad de generalización.

El conjunto de entrenamiento se destinó a la etapa de aprendizaje del modelo, ajustando los parámetros internos con base en las variables predictoras y la variable objetivo (weight_in_kg). El conjunto de prueba, equivalente al 20 % del total de los datos, se reservó exclusivamente para la evaluación final del modelo, sin intervenir en la fase de ajuste. La proporción 80/20 es una práctica ampliamente aceptada en problemas de regresión supervisada, pues equilibra adecuadamente la cantidad de datos disponibles para aprendizaje y validación de resultados.

En cuanto al conjunto de validación, es importante aclarar que no se generó una partición específica con este propósito. En las redes neuronales implementadas mediante scikit-learn, el proceso de validación puede gestionarse de forma interna durante el entrenamiento por ejemplo, a través de fracciones automáticas de los datos o técnicas de detención temprana (early stopping), sin requerir un conjunto separado. Por ello, en este proyecto se consideró suficiente la división tradicional en entrenamiento y prueba.

Finalmente, ambos subconjuntos fueron almacenados en el entorno de Google Drive, garantizando la disponibilidad, seguridad y trazabilidad de la información procesada a lo largo del flujo de trabajo.

4.3.3. Data Set de Imágenes.

De las imágenes disponibles en el conjunto de datos, se seleccionaron aquellas que muestran la vista posterior o trasera del animal como referencia principal para el análisis visual. Esta decisión

se fundamenta en criterios tradicionales de evaluación morfológica y zootécnica, ampliamente utilizados en la ganadería para estimar la condición corporal y el peso vivo de los ejemplares.

En la práctica ganadera, la observación de la parte posterior del animal —particularmente la región de la cadera, el anca y el tren posterior— permite identificar con mayor precisión el grado de desarrollo muscular y la proporción de masa corporal, indicadores directos del peso y estado físico. Estudios sobre evaluación fenotípica en bovinos señalan que la anchura de cadera, la forma del lomo y la cobertura de grasa en la grupa son variables morfológicas fuertemente relacionadas con la masa total del animal y su rendimiento productivo [30] [31] [32].

Por esta razón, la selección de imágenes traseras se consideró la más adecuada para representar la estructura corporal global y servir como insumo en procesos de estimación automática de peso mediante visión computacional. Además, esta perspectiva ofrece menor interferencia de otros elementos (como cabeza o extremidades) y mayor homogeneidad visual, lo que facilita la segmentación y el análisis posterior de la forma corporal.

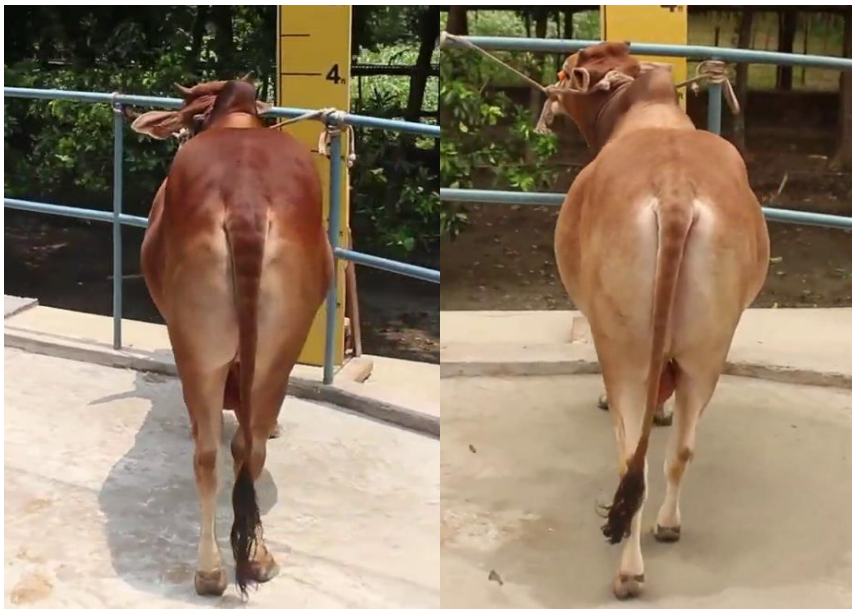


Figura 13: Ejemplos de imágenes seleccionadas desde la vista posterior, empleadas como referencia morfológica para la estimación del peso.

5. MODELOS INICIALES

5.1. Regresiones Iniciales

El objetivo de esta fase fue evaluar el desempeño de distintos modelos de regresión supervisada tradicionales, con el fin de establecer una línea base de comparación frente a los modelos de aprendizaje profundo implementados posteriormente.

Estos algoritmos se aplicaron al conjunto de datos pre procesado (513 registros con variables numéricas y categóricas codificadas) para estimar el peso corporal del animal (*weight_in_kg*) en función de sus características morfo métricas y categóricas.

Los modelos seleccionados fueron escogidos por su capacidad de representar diferentes enfoques de aprendizaje de aprendizaje supervisado, desde métodos lineales hasta no lineales y basados en ensambles.

Se implementaron modelos de regresión tradicionales como el SVR (Support Vector Regressor), bosque aleatorio (Random Forest), XGBoost (eXtreme Gradient Boosting) y MLP (Perceptrón Multicapa) con propósito de establecer una línea de base de desempeño. Estos modelos fueron entrenados con las variables normalizadas mediante StandardScaler, utilizando el 80% de los datos para entrenamiento y 20% para la prueba. Los resultados iniciales mostraron que los modelos basados en arboles (Random Forest y XGboost) ofrecieron mejor capacidad de generalización que los modelos lineales sirviendo como referencia para las pruebas posteriores con redes neuronales.

5.1.1. Modelos de ML Tradicionales.

La *figura 14* presenta la arquitectura general y el flujo de trabajo (pipeline) diseñado para el desarrollo de los modelos de machine learning: *SVR (Support Vector Regressor), Random Forest y XGBoost (eXtreme Gradient Boosting)*. Dicho esquema se orienta específicamente a la tarea de regresión abordada. El proceso metodológico se estructura en cuatro fases principales:

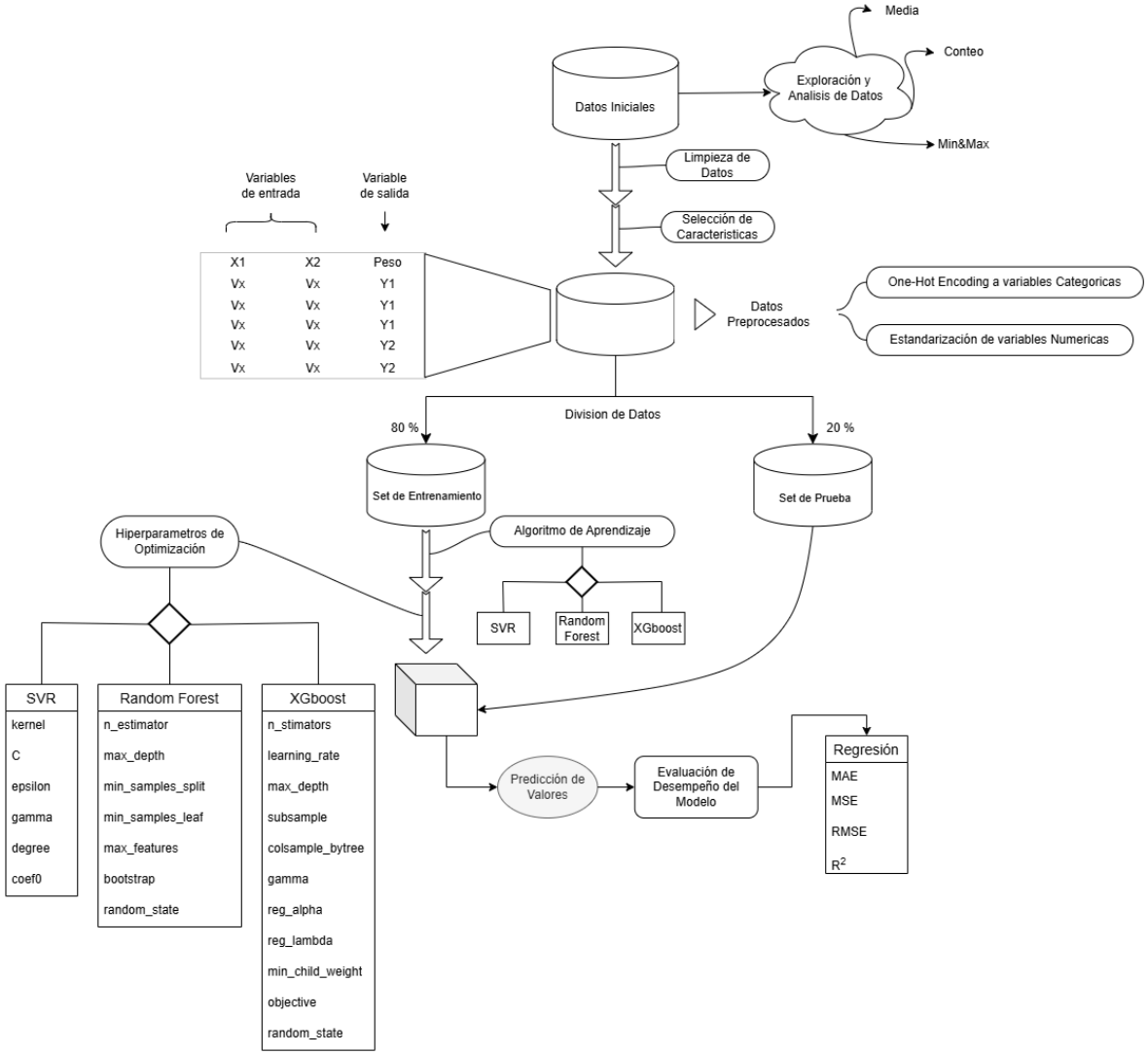


Figura 14: Diagrama desarrollo de Modelos Tradicionales.

A. Pre procesamiento y Exploración de Datos

El flujo se inicia con la ingesta de los Datos Iniciales. Inmediatamente, estos datos brutos pasan por dos procesos paralelos cruciales:

- *Exploración y Análisis de Datos (EDA):* Se realiza un análisis exploratorio para comprender la naturaleza de los datos, calculando métricas descriptivas fundamentales como la Media, el Conteo de registros, y los valores Min&Max (mínimos y máximos).
- *Limpieza de Datos:* Se aplican técnicas para manejar valores faltantes, atípicos o incorrectos.

Tras esta fase inicial, se realiza una Selección de Características para identificar las variables más relevantes para el modelo. Los datos resultantes se consolidan en un conjunto de Datos Preprocesados, el cual es sometido a transformaciones específicas:

- *One-Hot Encoding:* Aplicado a las variables Categóricas para convertirlas en un formato numérico que el modelo pueda interpretar.
- *Estandarización de variables Numéricas:* Se estandarizan las variables numéricas implementado el StandardScaler() para asegurar que todas tengan una escala comparable, mejorando la convergencia y el rendimiento de ciertos algoritmos.

B. Definición y División de Datos

Una vez pre procesado, el conjunto de datos se estructura formalmente, separando las Variables de entrada (caracteristicas o predictores, denotadas como X1, X2... V'x) de la Variable objetivo o el valor a predecir, denotado como 'Peso', Y1, Y2... Yn).

Posteriormente, este conjunto de datos se somete a una División de Datos estratégica:

- *Set de Entrenamiento (80%):* Una mayoría de los datos se reserva para entrenar los modelos.
- *Set de Prueba (20%):* Una porción menor, no vista por el modelo durante el entrenamiento, se reserva para su evaluación objetiva.

C. Entrenamiento y Optimización del Modelo

El Set de Entrenamiento se utiliza para alimentar el Algoritmo de Aprendizaje. Este proceso es guiado por un conjunto de Hiperparámetros de Optimización (como los que se usarían en una búsqueda en grilla o *GridSearchCV*).

El diagrama contempla la experimentación con tres algoritmos de aprendizaje supervisado para regresión, cada uno con sus propios hiperparámetros a optimizar:

- *SVR (Support Vector Regression):* Para el modelo SVR se realizó un proceso de ajuste de hiperparámetros mediante el método GridSearchCV, definido con la métrica de evaluación R^2 y orientado a maximizar el desempeño del modelo. A diferencia de la búsqueda aleatoria, este método evalúa exhaustivamente todas las combinaciones posibles dentro de los valores especificados para cada hiperparámetro.

  La Tabla 5 resume los rangos explorados para cada parámetro y muestra los valores óptimos encontrados tras completar la búsqueda en malla.

SUPPORT VECTOR REGRESSOR		
Herramienta: GridSearchCV (scoring = 'r2')		
Hiperparámetro	**Valores explorados**	**Mejor valor encontrado**
C	0.1, 1, 10, 50, 100, 300, 700, 1000	1000
epsilom	0.001, 0.01, 0.05, 0.1, 0.5	0.5
gamma	'scale', 'auto', 0.0005, 0.001, 0.005, 0.01, 0.1	'auto'

Tabla 5: Valores explorados y resultados de la búsqueda de parámetros para modelo SVR

- *Random Forest:* Para el modelo Random Forest Regressor también se llevó a cabo un proceso de ajuste de hiperparámetros empleando el método *RandomizedSearchCV,* con el objetivo de maximizar el valor del R^2. A diferencia de una búsqueda exhaustiva, este método selecciona de manera aleatoria combinaciones de valores dentro de rangos previamente definidos, lo que permite explorar el espacio de búsqueda de forma más eficiente.

  La Tabla 6 presenta los hiperparámetros considerados, los intervalos o listas de valores explorados y, finalmente, los mejores valores encontrados durante el proceso de optimización.

RANDOM FOREST REGRESSOR		
Herramienta: RandomizedSearchCV (scoring = 'r2')		
Hiperparámetro	**Valores explorados**	**Mejor valor encontrado**
n_estimators	randint(100,500)	413
max_depth	randit(5,50)	34
min_samples_split	randit(2,20)	2
min_samples_leaf	randit(1,10)	2
max_features	['auto', 'sqrt', 'log']	sqrt
bootstrap	[True, False]	False

Tabla 6: Valores explorados y resultados de la búsqueda de parámetros para modelo RF

- *XGBoostRegressor:* En la Tabla 7 se presentan los hiperparámetros explorados durante el proceso de optimización del modelo *XGBoostRegressor*, así como los mejores valores encontrados. Para esta etapa se empleó *RandomizedSearchCV*, utilizando como métrica de evaluación el R^2, con el objetivo de maximizar la capacidad predictiva del modelo. La tabla resume, para cada hiperparámetro, el rango o distribución de valores considerados en la búsqueda aleatoria y el valor óptimo obtenido tras el ajuste.

XGBOOST REGRESSOR		
Herramienta: RandomizedSearchCV (scoring = 'r2')		
Hiperparámetro	**Valores explorados**	**Mejor valor encontrado**
n_estimators	randint(200,800)	607
learning_rate	uniform(0.01,0.3)	0.25
max_depth	randit(3,15)	5
subsample	uniform(0.5,0.5) #0.5-1	0.75
colsample_bytree	uniform(0.5,0.5) #0.5-1	0.67
gamma	uniform(0,5)	1.46
min_child_weight	randit(1,10)	2.0

Tabla 7: Valores explorados y resultados de la búsqueda de parámetros para modelo XGBOOST REGRESSOR

D. Evaluación del Desempeño

Una vez que los modelos) ha sido entrenado, se utiliza para generar una Predicción de Valores.

Esta predicción se dirige a la etapa final de Evaluación de Desempeño del Modelo. En este bloque, las predicciones generadas por el modelo se comparan directamente contra los valores reales del Set de Prueba.

Dado que la tarea es de Regresión, el rendimiento se cuantifica utilizando métricas estándar:

- **MAE** (Mean Absolute Error o Error Medio Absoluto)
- **MSE** (Mean Squared Error o Error Cuadrático Medio)
- **RMSE** (Root Mean Squared Error o Raíz del Error Cuadrático Medio)
- **R^2** (Coeficiente de Determinación)

Este proceso de evaluación permite seleccionar de manera objetiva el modelo con mejor rendimiento de los tres probados (*SVR, Random Forest, XGBoost*) para la tarea específica. Los resultados de las evaluaciones antes y después de la búsqueda de mejores hiperparámetros se presentan y analizan en el capítulo 7.

5.1.2. Perceptrón Multicapa – MLP.

El diagrama de la *figura 15*, detalla la arquitectura del modelo de Red Neuronal Artificial de tipo Perceptrón Multicapa (MLP), que se alimenta del Set de Entrenamiento (el 80% de los datos preprocesados).

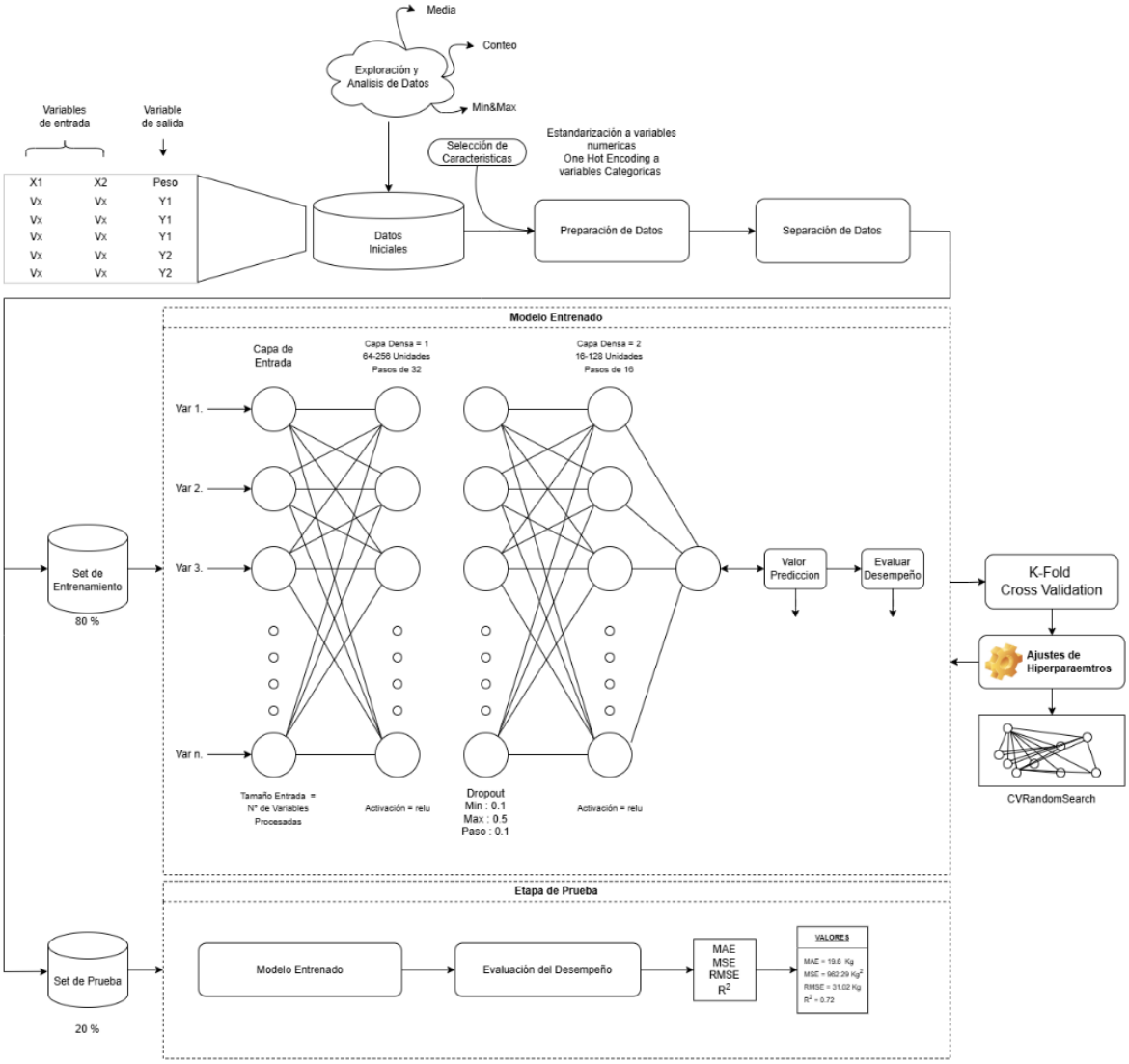


Figura 15: Diagrama del MLP

1. *Etapa de Pre procesamiento:* El flujo superior es consistente con la arquitectura general, mostrando los Datos Iniciales que pasan por Exploración, Selección de Características y Preparación de Datos (Estandarización y One-Hot Encoding) antes de dividirse en los conjuntos de entrenamiento (80%) y prueba (20%).

2. *Arquitectura del Modelo (MLP):* El Set de Entrenamiento se utiliza para entrenar una red neuronal con la siguiente topología para la búsqueda de parámetros:
    - *Capa de Entrada:* Recibe las 'n' variables de entrada (caracteristicas) resultantes del pre procesamiento.
    - *Capa Densa 1 (Oculta):* Definida en un rango de 64 a 256 neuronas con una paso de 32 y función de activación ReLU.
    - *Capa de Regularización (Dropout):* Se aplica un rango de 0.1 a 0.5 con un *paso* de 0.1 para prevenir el sobreajuste.
    - *Capa Densa 2 (Oculta):* Una segunda capa oculta de 16 a 128 neuronas, pasó de 16 y también con activación ReLU.
    - *Capa de Salida:* Genera un único valor con una sola neurona correspondiente al peso buscado.
3. *Proceso de Entrenamiento y Validación Interna:*
    - Durante el entrenamiento, el modelo calcula una función de LOSS (pérdida) para ajustar sus pesos (backpropagation).
    - Paralelamente, se Evalúa el Desempeño del modelo usando las métricas MAE, MSE, RMSE y R^2.
    - Crucialmente, este bloque de "Evaluar Desempeño" apunta a un proceso externo llamado *K-Fold Cross Validation*. Esto indica que la arquitectura MLP no se entrena de forma simple, sino que se valida utilizando esta técnica.
4. *Etapa de Prueba (Evaluación Final):*
    - Una vez que el modelo ha sido entrenado y optimizado (usando el proceso K-Fold que es descrito en la siguiente seccion), el Modelo Entrenado final se evalúa *una sola vez* contra el Set de Prueba (el 20% de los datos que se mantuvieron separados).
    - La Evaluación del Desempeño en esta etapa genera los resultados finales del proyecto que representan cómo se espera que el modelo funcione con datos completamente nuevos.

5.1.3. Estrategia de Validación (K-Fold-Cross Validation)

Este segundo diagrama mostrado en la *figura 16*, detalla exactamente qué ocurre en el bloque K-Fold Cross Validation mencionado en el diagrama del MLP. Esta es la estrategia de validación interna que se aplica *exclusivamente* al Set de Entrenamiento (el 80% de los datos).

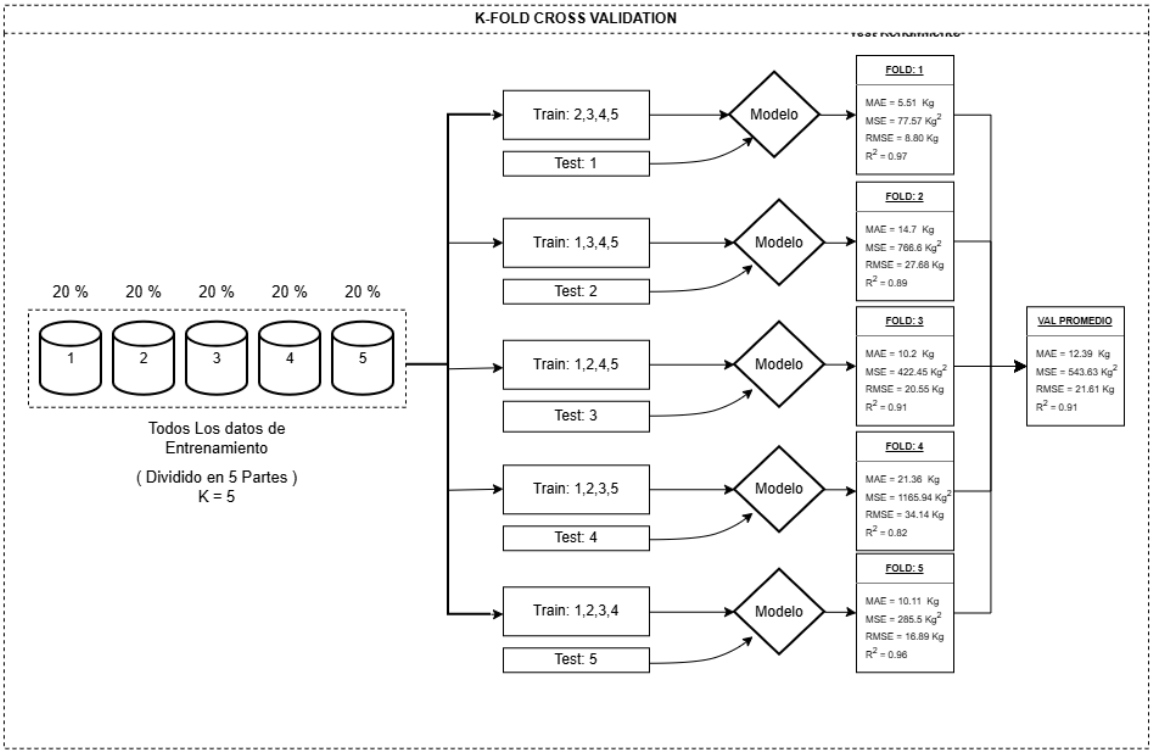


Figura 16: Diagrama de Validación Cruzada.

1. *División de Datos (K=5):* Todos Los datos de Entrenamiento se dividen en 5 partes (o "Folds") iguales, cada una conteniendo el 20% de estos datos.
2. *Proceso Iterativo:* El modelo (en este caso, la arquitectura MLP) se entrena y evalúa 5 veces (una por cada Fold):
    - *Fold 1:* El modelo se entrena con los Folds 2, 3, 4 y 5. Se valida (prueba) con el Fold 1. Se calculan las métricas (R^2=0.97).
    - *Fold 2:* El modelo se entrena con los Folds 1, 3, 4 y 5. Se valida con el Fold 2. Se calculan las métricas (R^2=0.86).
    - ... El proceso se repite para los Folds 3, 4 y 5.

3. *Resultado de la Validación:* Después de las 5 iteraciones, se obtiene un conjunto de 5 resultados para cada métrica. El rendimiento de validación del modelo se reporta como el VAL PROMEDIO (Valor Promedio) de esas 5 iteraciones.

   Para el modelo *Multi-Layer Perceptron Regressor (MLP*) se empleó un proceso de optimización de hiperparámetros mediante CVRandomSearch, utilizando como criterio de monitoreo la función de pérdida de validación (val_loss). Este procedimiento permitió explorar de manera eficiente diferentes configuraciones de arquitectura y entrenamiento, seleccionando aleatoriamente combinaciones dentro de los rangos definidos para cada hiperparámetro.

   La Tabla 8 presenta los valores explorados para cada parámetro y muestra los mejores resultados obtenidos tras completar la búsqueda, incluyendo el tamaño de las capas ocultas, tasa de dropout, tasa de aprendizaje y tamaño de lote.

MULTI LAYER PERCEPTRON REGRESSOR		
Herramienta: CVRandomSearch (monitor: val_loss)		
Hiperparámetro	**Valores explorados**	**Mejor valor encontrado**
Hidden_layer_size 1	min=64; max=256; step=32.	192
Hidden_layer_size 2	min=16; max=128; step=16.	16
dropout	min=0.1; max=0.5; step=0.1	0.4
learning_rate	min=1e-4;max=1e-2	0.00223423
batch_size	[16,32,64]	16

Tabla 8: Valores explorados y mejores resultados de hiperparámetros.

Se mantuvieron los mismos conjuntos de datos y las mismas métricas para todos los modelos con el fin de asegurar una comparación justa. Los resultados obtenidos y su análisis detallado se presentan en el capítulo 7.

6. MODELOS MULTIMODALES (HÍBRIDOS)

6.1. Modelo Multimodal Paralelo.

Para abordar el problema de predicción, se diseñó una arquitectura de red neuronal multimodal que procesa simultáneamente datos tabulares y datos de imágenes. A continuación, se detalla la estructura del modelo y se analizan sus resultados de entrenamiento.

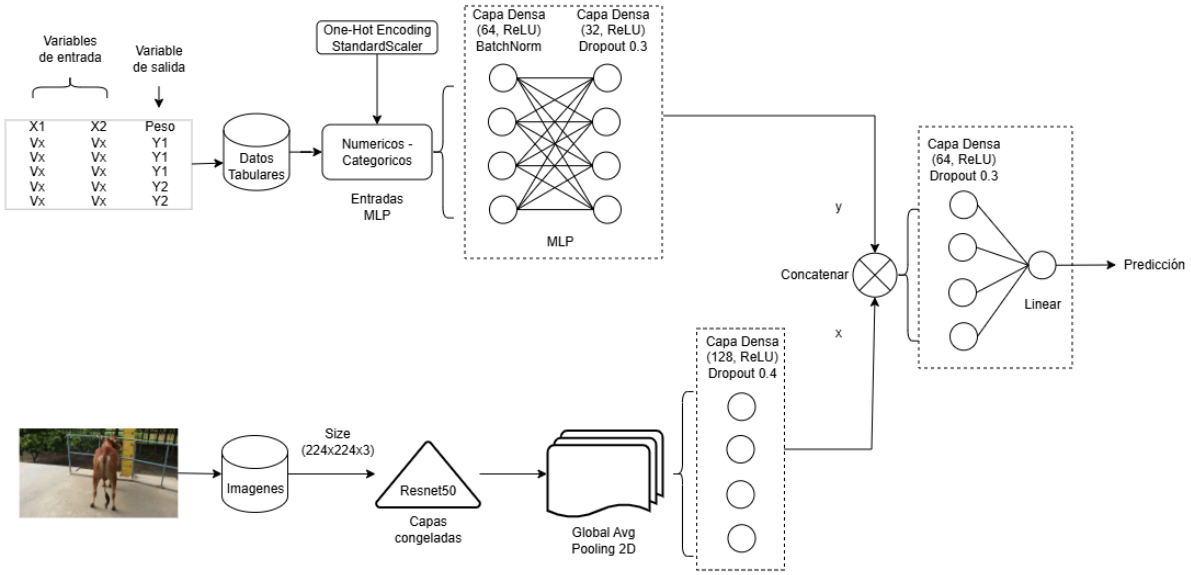


Figura 17: Diagrama de red multimodal o hibrida

Arquitectura del Modelo

El modelo que se ilustra en la *figura 17*, se compone de dos ramas de procesamiento paralelas que luego de fusionan para la predicción final.

- *Rama de Datos Tabulares (MLP)*

    1. *Pre procesamiento:* Las variables de entrada tabulares se dividen en numéricas y categóricas. Los datos numéricos se normalizan usando el estandarización (StandardScaler), mientras que los categóricos se codifican mediante One-Hot Encoding.
    2. *Red MLP:* La características procesadas alimentan un perceptron Multicapa (MLP) compuesto por una capa densa de 64 neuronas (activación ReLU), una capa de BatchNormalization, una segunda capa densa de 32 neuronas (ReLU) y una capa de

Dropout con una tasa de 0.3 para regularización. La salida de esta rama se denomina *y*.

- *Rama de Datos de Imágenes (CNN)*

    1. *Pre procesamiento:* Las imágenes de entrada se redimensionan al tamaño estándar de (224, 224, 3).
    2. *Extracción de Características:* Se utilizó un modelo ResNet50 pre-entrenado con sus capas convolucional congeladas (transfer learning). Esto permite aprovechar el conocimiento aprendido por el modelo en grandes conjuntos de datos de imágenes y aplicarlo a conjunto de datos pequeños.
    3. *Agrupación y Red:* La salida del ResNet50 pasa por una capa Global Average Pooling 2D para reducir la dimensionalidad. Posteriormente, se procesa con una capa densa de 128 neuronas (ReLU) seguida de un Dropout con tasa de 0.4. La salida de esta rama se denomina x.

- *Fusión y Predicción:*

    Las salidas de ambas ramas (*x* e *y*) se concatenan en un único vector de características. Este vector combinado alimenta la cabeza de predicción final, que consiste en una capa densa de 64 neuronas (ReLU), un Dropout de 0.3 y, finalmente, una *capa Lineal* que genera la predicción de salida (regresión).

Evaluación del Entrenamiento

El rendimiento del modelo durante el entrenamiento se evaluó utilizando el Error Cuadrático Medio (MSE) como función de pérdida principal y el Error Absoluto Medio (MAE) como métrica de interpretabilidad. Las gráficas "Evolución del Loss (MSE)" y "Evolución del MAE" de la figura 18, muestran el comportamiento del modelo durante 15 épocas.

- *Convergencia del Modelo:* Ambas gráficas muestran una convergencia rápida durante las primeras 2-3 épocas. El *Train Loss* (MSE) y el *Train MAE* disminuyen drásticamente desde sus valores iniciales (>60,000 y ~250, respectivamente), lo que indica que el modelo aprende rápidamente los patrones fundamentales de los datos.

- *Estabilidad y Generalización:* Después de la tercera época, las curvas de pérdida y MAE (tanto de entrenamiento como de validación) se estabilizan, mostrando una mejora más gradual y estable.
- *Análisis de Sobreajuste (Overfitting):* El aspecto más destacable de las gráficas es la proximidad entre las curvas de entrenamiento y validación (*Train Loss* vs. *Val Loss*, y *Train MAE* vs. *Val MAE*). La brecha entre ellas es mínima y se mantiene constante a lo largo del entrenamiento.

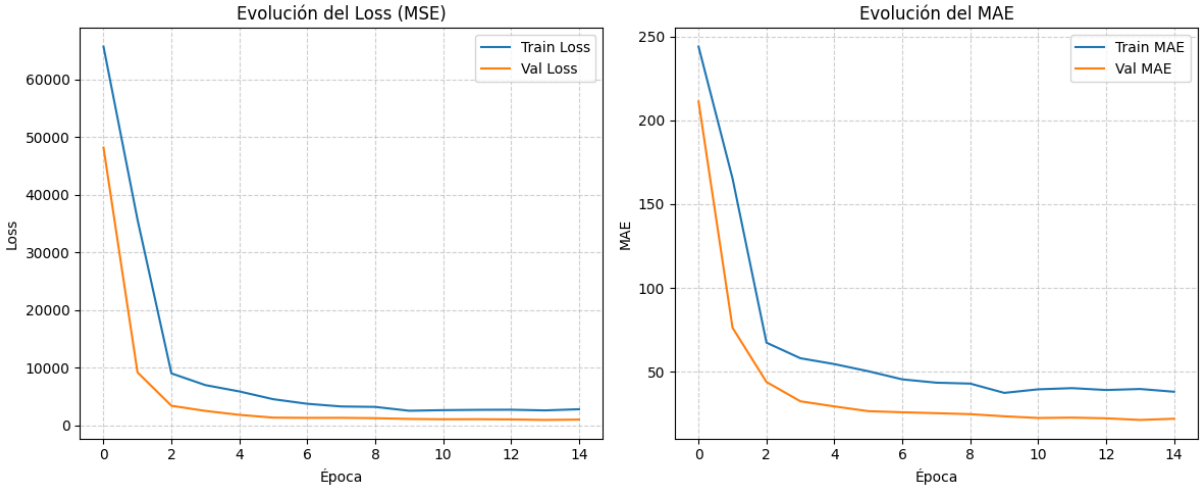


Figura 18: Evolución del Loss (MSE) y Evolución del MAE.

La arquitectura multimodal implementada demuestro ser efectiva. El análisis de las curvas de entrenamiento y validación confirmaron que el modelo generaliza correctamente a datos no vistos. La ausencia de una brecha significativa entre las métricas de entrenamiento y validación indica que las técnicas de regularización empleadas (congelamiento de capas en ResNet50, Dropout en múltiples puntos y BatchNormalization) fueron exitosas para prevenir el sobreajuste (overfitting). El modelo convergió de manera estable a una solución robusta.

6.2. Modelo Multimodal en Serie

La metodología adoptada (mostrada en la figura 19) corresponde también a un enfoque híbrido que integra técnicas de visión por computador basadas en aprendizaje profundo con algoritmos de aprendizaje automático tradicional. El objetivo fue estimar el peso del animal a partir de imágenes RGB, para lo cual se diseñó un pipeline compuesto por dos etapas principales:

1. Obtención automática de características mediante una red convolucional (VGG16) ajustada parcialmente mediante fine-tuning, seguido de una
2. Predicción del peso mediante un modelo de regresión basado en Random Forest.

La separación entre la fase de extracción de características y la fase de regresión aporto versatilidad y permitio un mayor control en el análisis de resultados.

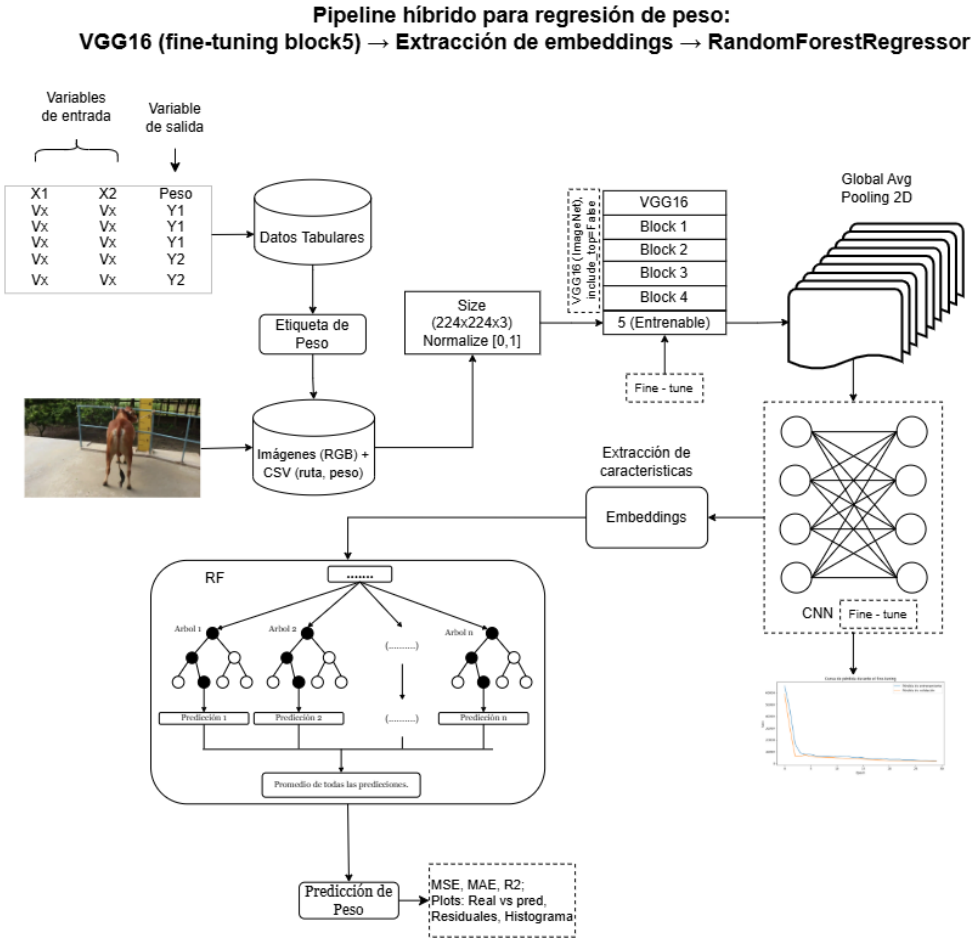


Figura 19: Diagrama de flujo modelo multimodal en serie CNN + RF

6.2.1. Conjunto de datos

El conjunto de datos utilizado también fue compuesto por las imágenes RGB de los animales y el archivo CSV asociado que incluye las rutas a cada imagen y la variable objetivo: **peso del animal (kg)**.

El conjunto de datos tabulares fue dividido en tres subconjuntos mutuamente excluyentes y almacenados en el Google Drive.

- *Entrenamiento (train):* utilizado para ajustar los parámetros del modelo.
- *Validación (val):* empleado para seleccionar hiperparámetros y monitorear sobreajuste.
- *Prueba (test):* reservado exclusivamente para la evaluación final del sistema.

Esta división garantiza una evaluación objetiva del desempeño.

6.2.2. Preprocesamiento y pipeline de datos

Para garantizar la compatibilidad con la arquitectura VGG16 y mejorar la eficiencia del entrenamiento, se aplicaron las siguientes transformaciones:

1. *Redimensionamiento de imágenes:* todas las imágenes se ajustaron a 224×224×3**,** tamaño estándar de entrada para VGG16.
2. *Normalización:* los valores de los píxeles se escalaron al rango **[0,1]**, lo que contribuye a la estabilidad numérica durante la propagación hacia adelante y hacia atrás.
3. *Pipeline tf.data:* se empleó un pipeline eficiente que incluye lectura secuencial, batching, shuffle y prefetch, optimizando así el uso de memoria y GPU.
4. *Data augmentation (solo para entrenamiento):* se aplicaron transformaciones no determinísticas (como inversión horizontal) con el fin de aumentar la variabilidad de las muestras y mitigar el riesgo de sobreajuste.

Este pre procesamiento permitió homogenizar la entrada y reducir la sensibilidad a variaciones no relevantes en las imágenes.

6.2.3. Modelo de extracción de características: VGG16 con Fine-Tuning

Para la rama de visión artificial del modelo multimodal en serie, se implementó la arquitectura VGG16, pre-entrenada en el extenso conjunto de datos ImageNet. Esta estrategia de Transferencia de Aprendizaje se adoptó para aprovechar las jerarquías de características visuales ya aprendidas, mitigando el riesgo de sobreajuste con el tamaño limitado del conjunto de datos bovino y optimizando el tiempo de entrenamiento.

La arquitectura VGG16 se cargó sin las capas densas finales (include_top=False), conservando únicamente los bloques convolucionales. Se implementó una estrategia de Fine-Tuning parcial y controlado para adaptar eficientemente los pesos del modelo al dominio zootécnico:

- *Congelamiento de Capas Inferiores (Bloques 1 a 4):* Los bloques convolucionales 1 a 4 se mantuvieron congelados (layer.trainable = False). Esto aseguró la preservación de los filtros de bajo nivel (ej. detección de bordes y texturas), considerados genéricos y universales para la visión por computador, maximizando la retención del conocimiento previo de ImageNet.
- *Descongelamiento y Especialización (Bloque 5):* Únicamente el Bloque 5 (el bloque convolucional más profundo) se habilitó como entrenable (layer.trainable = True). Esta decisión fue crucial, ya que permite que los filtros más abstractos de la red se especialicen en la detección de características de alto nivel específicas del ganado, como la forma de la grupa, el perímetro y los contornos de volumen, que son directamente correlacionables con el peso.

Tasa de Aprendizaje Diferencial y Optimización

Para gestionar el Fine-Tuning de manera segura, se utilizó el optimizador ADAM con una tasa de aprendizaje diferencial (1×10^{-5}):

Implementación: Se configuró el optimizador con una tasa de aprendizaje base de 1×10^{-5} (learning_rate=$1e^{-5}$) aplicada a todas las capas entrenables (el Bloque 5 y la cabeza de regresión añadida).

Justificación: Esta tasa de aprendizaje muy baja se seleccionó específicamente para evitar que los grandes gradientes generados durante las primeras épocas de entrenamiento sobrescribieran o "destruyeran" los pesos pre-entrenados del Bloque 5. Al aplicar un factor de aprendizaje lento y gradual, se aseguró un ajuste fino y controlado que permitió a la red refinar las características para la regresión del peso, sin desestabilizar la base del modelo.

Esta estrategia aseguró que la red VGG16 funcionara como un detector de características morfo métricas altamente especializadas, siendo el input ideal para la etapa posterior de regresión.

6.2.4. Cabeza del modelo y función de pérdida

Posteriormente, se agregó una capa de Global Average Pooling 2D (GAP), con el fin de transformar el mapa de activaciones en un vector unidimensional compacto. Esta representación constituye el *embedding* utilizado para la fase posterior de regresión.

El modelo se entrenó utilizando:

- *Función de pérdida:* MSE (Mean Squared Error)
- *Optimizador:* Adam
- *Métricas complementarias:* MAE, RMSE y R^2
- *Callbacks:* EarlyStopping, ReduceLROnPlateau y ModelCheckpoint

Estos elementos favorecen la convergencia del modelo y la selección automática de hiperparámetros durante el proceso de entrenamiento.

6.2.5. Extracción de embeddings.

Una vez entrenado y ajustado el modelo VGG16, se creó un feature extractor, es decir, un modelo auxiliar cuyo propósito es recibir una imagen y retornar únicamente el vector de características generado por la capa GAP (Global Average Pooling).

Cada imagen de los subconjuntos de entrenamiento, validación y prueba fue propagada a través del extractor

Estos vectores son representaciones densas que codifican patrones visuales de alto nivel como proporciones, formas, contornos y texturas relevantes para la estimación del peso.

6.2.6. Modelo de regresión basado en RandomForest

Para la predicción final del peso se utilizó un RandomForestRegressor, con 200 árboles de decisión entrenados sobre subconjuntos aleatorios de características y datos.

Las razones principales para seleccionar este modelo son:

- Alta robustez frente a ruido y datos no lineales.
- Reducción de la varianza mediante el promedio de múltiples predictores.
- Facilidad para interpretar la importancia relativa de las características.
- Compatibilidad con embeddings de alta dimensión sin necesidad de escalamiento adicional.

El modelo de Random Forest produce la predicción del peso a partir del promedio ponderado de las predicciones individuales de cada árbol, lo cual proporciona estabilidad y precisión.

6.2.7. Evaluación del sistema

El desempeño final del pipeline se evaluó mediante métricas ampliamente utilizadas en problemas de regresión:

- Mean Squared Error (MSE)
- Mean Absolute Error (MAE)
- Coeficiente de determinación (R^2)

Además de estas métricas, se realizaron análisis visuales para una comprensión más profunda del comportamiento del sistema:

- Gráficos de valores reales vs. predichos
- Histogramas de error
- Curvas de entrenamiento de la CNN (loss y val_loss)

Estas herramientas permitieron validar el comportamiento del pipeline tanto cuantitativa como cualitativamente.

6.2.8. Integración del pipeline completo

El flujo final del sistema puede describirse en cuatro fases:

1. *Pre procesamiento:* normalización y redimensionamiento de las imágenes.
2. *Extracción de características:* propagación de las imágenes a través de VGG16 con fine-tuning del bloque 5.
3. *Generación de embeddings:* obtención del vector representativo mediante la capa GAP.
4. *Predicción:* utilización de RandomForestRegressor para estimar el peso final.

Este pipeline aprovecha de manera eficiente la combinación de modelos profundos y técnicas tradicionales de regresión, logrando una arquitectura más interpretable, modular y flexible.

7. ANÁLISIS DE RESULTADOS

7.1. Resultados

Modelo	Tipo de Datos	Resultados Modelo Base	Resultados Modelo Mejorado
SUPPORT VECTOR REGRESSOR	**Tabulares**	MAE:11.22 kg MSE:1236.18 kg² RMSE:35.15 kg R2: 0.65	MAE:7.40 kg MSE:377.9 kg² RMSE:19.4 kg R²:0.89
RANDOM FOREST REGRESSOR		MAE: 4.01 kg MSE: 76.90 kg² RMSE: 8.77 kg R²: 0.978	MAE:11.67 kg MSE:468.39 kg² RMSE: 21.64 kg R²: 0.86
XGBOOST REGRESSOR		MAE: 4.07 kg MSE: 84.14 kg² RMSE: 9.17 kg R²: 0.976	MAE: 3.27 kg MSE: 33.71 kg² RMSE: 5.80 kg R²: 0.990
MLP REGRESSOR		MAE: 20.03 kg MSE: 951.09 kg² RMSE: 30.83 kg R²: 0.73	MAE:14.58 kg MSE:612.91 kg² RMSE:24.75 kg R²: 0.82

Tabla 9: Resultados Modelos Tradicionales

Modelo	Tipo de Datos	Resultados Modelo Base	Resultados Modelo Mejorado (fine-tuning)
MULTIMODAL PARALELO	**Tabulares e Imágenes**	MAE: 35.00 kg MSE:1880.64 kg² RMSE:43.37 kg R² : 0.47 kg	MAE:21.57 kg MSE:916.85 kg² RMSE:30.28 R² :0.743
MULTIMODAL SERIE	**Imágenes**	MAE : 46.43 kg MSE : 4957.81 kg² RMSE: 70.41 kg R² : -0.11	MAE: 29.91 kg MSE: 1754.20 kg² RMSE:41.88 kg R² :0.60

Tabla 10: Resultados Modelos Multimodales

7.2 Análisis de Resultados

La Tabla 9, correspondiente a los modelos tradicionales tanto en su versión base como después de la optimización de hiperparámetros, muestra que estos enfoques alcanzan un desempeño sobresaliente en la tarea de predicción. Este buen rendimiento puede explicarse por la riqueza y cantidad de variables tabulares disponibles durante el entrenamiento. Dentro de este grupo, el modelo XGBoost Regressor se posiciona como el de mayor precisión tras la etapa de mejora, mientras que Random Forest destaca como el mejor desempeño en su versión base.

Dado que la variable objetivo se encuentra entre 150 y 816 kg, un MAE cercano a 3.2 kg representa un error medio inferior al 1 % del rango total, lo que evidencia una altísima precisión y una excelente capacidad de generalización. En particular, el modelo XGBoost Regressor muestra el mejor desempeño global, con valores de MAE = 3.27 kg, RMSE = 5.8 kg, MSE = 33.71 kg² y un R^2 de 0.99, indicando que explica aproximadamente el 99 % de la variabilidad de los datos. Este rendimiento supera ligeramente al obtenido por el modelo base Random Forest Regressor, que alcanza MAE = 4.01 kg, RMSE = 8.7 kg, MSE = 76.9 kg² y un $R^2 = 0.976$.

Por el contrario, modelos como SVR (Support Vector Regressor) y MLP (Perceptrón Multicapa) mostraron errores más elevados, con MAE de 7.4 kg y 14.8 kg, respectivamente en los modelos mejorados. Esto se ve reflejado también en sus valores de R^2 (0.89 y 0.82), que indican una menor capacidad para explicar la variabilidad de los datos en comparación con Random Forest y XGBoost, justificando así su desempeño inferior.

En el caso de los modelos multimodales (híbridos), los resultados presentados en la Tabla 10 (mejora lograda mediante *fine-tuning* de redes pre entrenadas respecto a una configuración base inicial con una CNN clásica) muestran que la integración de imágenes y datos tabulares no logró superar el desempeño alcanzado por los modelos tradicionales basados exclusivamente en variables tabulares. En particular, el modelo multimodal en configuración paralela obtuvo un MAE de 21.57 kg, un RMSE de 30.28 kg y un coeficiente de determinación R^2 de 0.74, lo que evidencia que el modelo fue capaz de aprender una relación coherente y estable entre ambas modalidades, aunque sin alcanzar la precisión de los enfoques puramente tabulares.

El MAE obtenido representa un error aproximado del 3.1% respecto al rango total de la variable objetivo, lo cual refleja una buena capacidad de aproximación pese a la complejidad del problema. Por su parte, un R^2 del 74% indica que el modelo explica gran parte de la variabilidad de los datos, aunque aún existe un 26% no capturado, lo cual sugiere que su desempeño es adecuado pero mejorable, sin que esto implique que el modelo sea ineficiente o inválido.

En el caso de Random Forest se observa un comportamiento inusual: el modelo base supera al modelo ajustado, mostrando un MAE menor. Esto puede deberse a que la búsqueda de hiperparámetros exploró combinaciones que, aunque óptimas en validación, redujeron la capacidad

de generalización en prueba. Parámetros demasiado restrictivos como pocos árboles, profundidad limitada o mínimos de muestras elevados pueden generar sub ajuste.

En conjunto, estos factores explican por qué el modelo "mejorado" puede rendir peor que la base. Esto resalta la necesidad de validaciones más robustas, como validación cruzada o múltiples búsquedas con distintas semillas, para asegurar mejoras reales en la capacidad predictiva.

En el caso del modelo multimodal en serie, que utiliza únicamente información visual, se observó un desempeño ligeramente superior al del modelo multimodal en paralelo, alcanzando un MAE de 29.9 kg, un RMSE de 1 754 y un R^2 de 0.6. Aunque un R^2 del 60 % indica que el modelo logra captar cierta proporción de la variabilidad presente en los datos, su desempeño continúa siendo marcadamente inferior al alcanzado por los modelos basados en datos tabulares y por el modelo multimodal en paralelo. Esto sugiere que, en este caso, la información proveniente de las imágenes por sí sola no es suficiente para realizar predicciones con alta precisión.

En términos generales, los resultados permiten concluir que los modelos multimodales evaluados no lograron integrarse de manera eficiente, y ninguno de ellos alcanzó el nivel de precisión obtenido por los modelos construidos únicamente con variables tabulares. Esto evidencia que la información estructurada posee un mayor poder predictivo en este problema específico y que la fusión de modalidades no aportó mejoras significativas en el desempeño.

7.2.1. Resultados del Modelo Multimodal en Serie

El modelo multimodal en serie fue seleccionado para su implementación en la aplicación prototipo debido a que su entrenamiento se realizó exclusivamente con imágenes. Esto permite que, al cargar una fotografía desde la cámara o la galería del dispositivo, el sistema genere una predicción sin requerir información tabular adicional por parte del usuario.

Por esta razón, se presentan a continuación resultados adicionales que permiten un análisis más completo del desempeño del modelo.

La *figura 20*, del gráfico Real vs. Predicción muestra que existe una tendencia positiva entre los valores reales y las predicciones, lo que indica que el modelo Random Forest logra capturar parcialmente la relación entre las características de entrada y el valor objetivo. Sin embargo, para valores bajos y medios (entre 200 y 300), el modelo tiende a agrupar sus predicciones en un rango

estrecho, con ligera dispersión alrededor de la línea ideal, evidenciando un comportamiento conservador que evita valores extremos y concentra los resultados cerca de la media del conjunto. En contraste, para valores altos (por encima de 400) se observa un sub ajuste evidente, ya que el modelo predice valores sistemáticamente menores que los reales; esto se refleja en la acumulación de puntos por debajo de la línea de referencia. Finalmente, la dispersión general evidencia que el error no es uniforme, pues algunos casos presentan desviaciones notables tanto por encima como por debajo de los valores observados, lo que sugiere limitaciones en la capacidad del modelo para generalizar adecuadamente en todo el rango de datos.

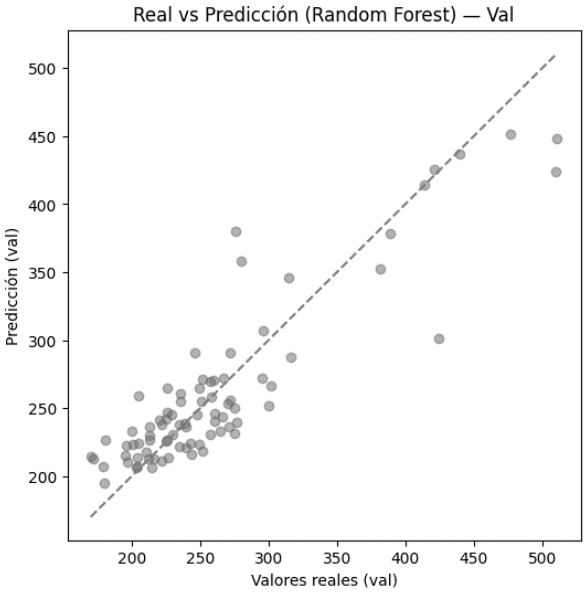


Figura 20: Gráfico de Valores Reales Vs Predicciones

El histograma de residuales (mostrado en la *figura 21*) del conjunto de validación muestra que la mayoría de los errores del modelo Random Forest se concentran alrededor de cero, lo cual indica que, en términos generales, las predicciones tienden a aproximarse a los valores reales. La distribución presenta una forma aproximadamente simétrica, aunque con cierta dispersión hacia ambos extremos, lo que refleja la presencia de errores tanto positivos como negativos. No obstante, se observan valores residuales más alejados del centro, especialmente por debajo de –50 y por encima de 50, lo que evidencia la existencia de casos puntuales en los que el modelo presenta desviaciones significativas. Estos valores atípicos sugieren que el modelo no logra capturar adecuadamente ciertos patrones específicos del conjunto de validación, lo cual coincide con el comportamiento observado previamente en el análisis del gráfico Real vs. Predicción. En conjunto,

el histograma revela un desempeño razonable en la mayoría de los casos, pero también pone en evidencia limitaciones en la capacidad del modelo para generalizar en situaciones menos comunes o de mayor complejidad.

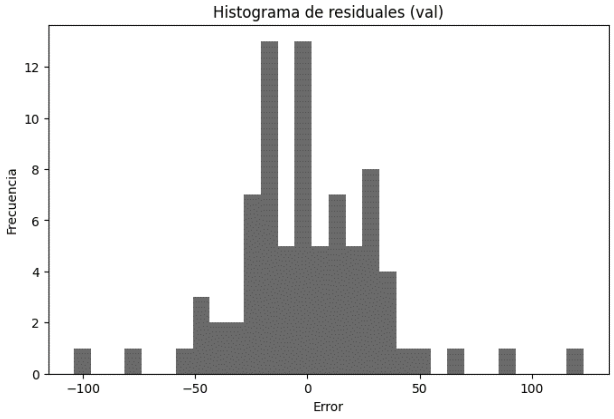


Figura 21: Histograma de Residuales

La *figura 22* muestra la evolución de la función de pérdida durante el proceso de *fine-tuning* del modelo en el entrenamiento de la CNN en el bloque 5 de la red VGG16. En ella se observa tanto la pérdida de entrenamiento como la pérdida de validación a lo largo de 30 épocas. Inicialmente, ambas curvas presentan valores muy altos, lo que indica un error considerable al inicio del ajuste fino.

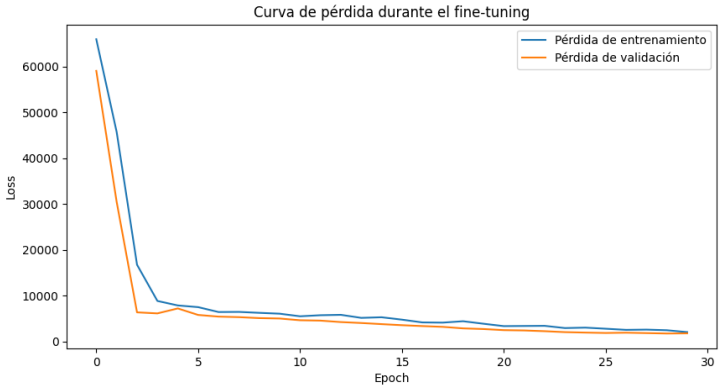


Figura 22: Curva Loss del Fine-Tuning

A medida que avanzan las épocas, ambas pérdidas descienden de manera pronunciada durante las primeras iteraciones, lo que evidencia que el modelo está aprendió rápidamente los patrones relevantes de los las imágenes. Posteriormente, el descenso se vuelve más gradual, estabilizándose

alrededor de valores bajos y similares para ambas curvas, lo cual sugiere que no se presenta sobreajuste significativo.

La cercanía entre la pérdida de entrenamiento y la de validación durante todo el proceso es un indicador positivo de la capacidad de generalización del modelo, demostrando que el *fine-tuning* logró ajustar adecuadamente los pesos sin comprometer el desempeño en datos no vistos.

8. LA APLICACIÓN MÓVIL PROTOTIPO

Con el fin de validar la funcionalidad del modelo en condiciones reales y garantizar su operatividad sin depender de infraestructura externa, se desarrolló una aplicación móvil para dispositivos Android, cuya ejecución se realiza completamente de manera local, sin procesamiento en la nube. Esta decisión se fundamenta en varias razones:

i. Evita el uso de API´s de pago que incrementarían los costos de operación del prototipo.
ii. Eliminar la dependencia de conectividad a internet, especialmente en contextos rurales donde la señal puede ser limitada o inexistente: y
iii. Asegurar que los usuarios tengan acceso permanente al sistema, aun cuando no haya disponibilidad de servidores externos.

Bajo este enfoque, fue necesario adaptar los modelos de Machine Learning descritos en el capítulo 6, para que pudieran ejecutarse de forma eficiente dentro del propio dispositivo móvil, funcionando como un verdadero edge device.

A continuación se detallan las tecnologías utilizadas y el proceso de desarrollo del aplicativo.

8.1 Tecnología y Arquitectura

El desarrollo del prototipo de la aplicación móvil se fundamentó en los principios de *Edge Computing*, buscando maximizar la eficiencia y la operatividad *offline* directamente en el dispositivo. Esta decisión fue crítica para garantizar la usabilidad en entornos rurales con conectividad limitada.

Las principales tecnologías de desarrollo empleadas fueron:

- *Android Studio (con Kotlin/Java):* Entorno de desarrollo para la construcción de la interfaz de usuario y la gestión de la lógica de negocio.
- *TensorFlow Lite (TFLite):* Framework de inferencia móvil de Google, esencial para la ejecución optimizada de la Red Neuronal Convolucional (CNN) en dispositivos con recursos limitados.

8.1.1. Arquitectura de Despliegue del Modelo Híbrido

Se optó por desplegar la versión del modelo multimodal en serie, el cual utiliza exclusivamente la entrada visual, cumpliendo con el objetivo de estimar el peso a partir de una única imagen sin requerir datos tabulares adicionales del usuario. La arquitectura se implementó en dos etapas secuenciales:

Extracción de Características Visuales (TensorFlow Lite)

Esta etapa maneja el procesamiento de la imagen y se enfoca en la eficiencia y el tamaño del modelo:

- *Modelo Base:* Se emplea la red convolucional VGG16, pre-entrenada para extraer un vector de características (*embeddings*) a partir de las imágenes de entrada.
- *Optimización para Borde*: La red VGG16, entrenada inicialmente en Python (TensorFlow), fue convertida al formato TensorFlow Lite (.tflite). Este proceso incluyó una Compresión de pesos de los pesos del modelo, lo que redujo significativamente su peso en disco y aceleró la velocidad de la inferencia en la Unidad de Procesamiento del dispositivo móvil.

Regresión Final (Módulos Nativos de Java)

Los *caracteristicas* generados por la VGG16 son procesados por el modelo final de regresión:

- *Conversión a Código Nativo:* El modelo Random Forest (RF), entrenado en Python, fue exportado y convertido a código nativo Java. Esto era necesario debido a que el *back-end* de la aplicación Android se ejecuta sobre la Máquina Virtual de Java (JVM).
- *Modularización del Modelo (Solución de Ingeniería):* La conversión de un modelo Random Forest con una alta cantidad de árboles genera una extensa estructura de código basada en sentencias *if-else*. El archivo de código Java resultante excedía los límites de tamaño de método impuestos por el sistema operativo Android, impidiendo su ejecución. Para resolver este problema de ingeniería, el modelo fue estratégicamente fragmentado y desplegado en cinco módulos independientes. Un controlador principal orquesta la ejecución de estos sub módulos, permitiendo que la lógica de predicción sea idéntica a la entrenada en el ambiente de ciencia de datos, pero viable y eficiente en el entorno móvil.

Pipeline de Procesamiento en Tiempo Real

Esta secuencia completa se implementó en la actividad principal (MainActivity). El *pipeline* se ejecuta *offline* e incluye:

1. *Adquisición:* Captura o selección de la imagen.
2. *Pre procesamiento:* Escalado, normalización de píxeles y formateo para ajustarse a los requisitos del modelo (ej. 224x224 píxeles).
3. *Inferencia CNN:* La imagen pasa al modelo TFLite para la extracción del vector de características.
4. *Inferencia Regresión:* El vector de características es procesado por los cinco módulos nativos de Random Forest para generar la estimación final del peso.

8.2. Flujo de Usuario y Funcionalidad

La aplicación desarrollada está diseñada para estimar el peso de un animal a partir de una imagen capturada o seleccionada por el usuario. Su interfaz gráfica se caracteriza por ser minimalista, clara y centrada en la usabilidad, permitiendo una interacción intuitiva incluso para usuarios sin experiencia técnica.

Pantalla inicial: interfaz antes de cargar una imagen

En la pantalla inicial (Ver figura 23), el usuario observa un fondo oscuro que sirve como base para resaltar los elementos interactivos. En la zona central se encuentra un espacio destinado a la futura visualización de la fotografía, aunque inicialmente aparece vacío. Justo debajo se despliega el texto *"Peso Estimado:"*, sin valores numéricos asociados, indicando que aún no se ha realizado ninguna predicción.

En la parte inferior se encuentran dos botones de gran tamaño, color verde y bordes redondeados:

- **"Tomar Foto"** Permite acceder a la cámara del dispositivo para capturar una nueva imagen del animal en tiempo real.
- **"Ir a Galería"** Abre el explorador multimedia del dispositivo para seleccionar una fotografía previamente almacenada.

Estos botones representan los puntos de entrada al sistema de inferencia y guían al usuario en el flujo de uso.

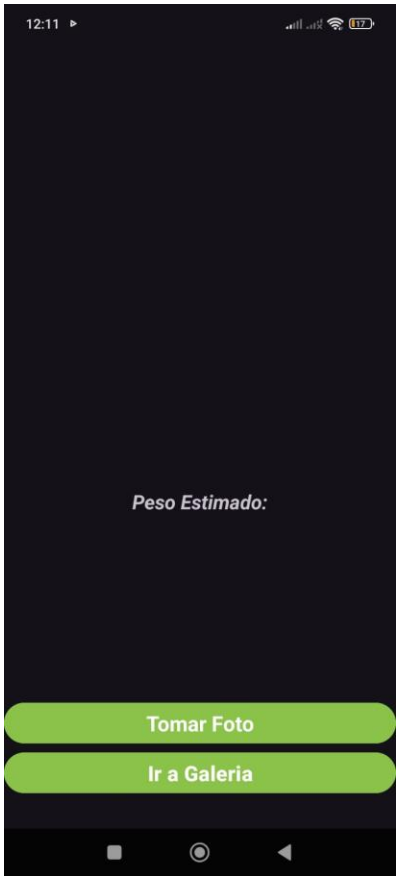


Figura 23: Interfaz Inicial

Pantalla tras cargar una imagen: interfaz con predicción

Una vez el usuario toma o selecciona una fotografía, la aplicación actualiza la vista para mostrar el resultado del procesamiento. La imagen del animal se visualiza en la parte superior, ocupando un espacio central prominente que permite verificar que la fotografía sea adecuada antes de proceder con otra captura si es necesario.

Debajo de la imagen, se presenta nuevamente el texto *"Peso Estimado:"*, pero ahora acompañado del valor predicho por el modelo de inteligencia artificial. Este valor aparece destacado en color rojo (por ejemplo, 248.17 kg**),** lo que facilita su lectura y señala claramente el resultado de la estimación. El diseño mantiene coherencia visual con la pantalla inicial, pero añade la información clave generada por el sistema.

Los botones **"Tomar Foto"** e **"Ir a Galería"** permanecen activos en la parte inferior, permitiendo repetir el proceso cuantas veces sea necesario sin navegar a otras ventanas.

Figura 24: Resultado Obtenido.

Flujo de funcionamiento general

1. El usuario abre la aplicación y se encuentra con la pantalla inicial.
2. Selecciona una imagen mediante cámara o galería.
3. La aplicación carga la fotografía y la procesa mediante un modelo de predicción previamente entrenado.
4. El resultado del peso estimado se muestra en pantalla, resaltado en color rojo.
5. El usuario puede repetir el proceso capturando otra foto o eligiendo un nuevo archivo desde la galería.

Enfoque de diseño

El diseño combina simplicidad visual con funcionalidad directa. El uso de un fondo oscuro mejora el contraste con la imagen y el texto; los botones verdes proporcionan un punto de atención claro y amigable; y la separación entre la imagen, el resultado y los controles garantiza una experiencia de usuario limpia y comprensible.

En conjunto, la interfaz soporta de forma eficiente la lógica principal de la aplicación: capturar o seleccionar una imagen, procesarla mediante el modelo de IA y mostrar el peso estimado de manera clara y accesible.

8.3. Evaluación Técnica y de Usabilidad del Prototipo

El desarrollo del prototipo móvil demostró la viabilidad técnica de portar arquitecturas híbridas complejas (CNN + Machine Learning Clásico) a entornos de *Edge Computing*. La integración de TensorFlow Lite para la extracción de características visuales y módulos nativos de Java para la regresión permitió realizar inferencias totalmente desconectadas (offline), cumpliendo con el requisito de operar en zonas rurales sin acceso a internet.

En términos de usabilidad, la interfaz minimalista redujo la fricción para el usuario final, permitiendo obtener una estimación de peso en menos de tres pasos. Sin embargo, las pruebas de campo preliminares evidenciaron que la precisión del sistema es altamente sensible a la calidad de la fotografía: factores como la iluminación, el ángulo de captura y la distancia al animal influyen en el vector de características generado por la VGG16.

Se concluye que, si bien la aplicación es funcional y operativa, su rol actual es el de una herramienta de apoyo para una estimación rápida y aproximada. Para convertirla en un instrumento de precisión comercial, será necesario implementar controles de validación de imagen en tiempo real que alerten al usuario si la foto no cumple con los estándares de calidad requeridos antes de procesar la predicción.

En la *tabla 11* y *figura 25* se detallan datos de desempeño de respuesta de la aplicación al realizar la estimación así como tamaño que ocupa la aplicación en el dispositivo web y lo que se logró reducir en el transformado de la red neuronal entrenada al formato TensorFlow Lite.

Métrica de Desempeño	Valor Estimado (Ajuste su valor real)	Justificación
Tiempo de Inferencia (Latencia)	1-2 segundos	Tiempo promedio desde la captura/selección hasta la estimación final. En un dispositivo Android Xiaomi Red Note 12.
Tamaño del APK (Binario)	44.32 MB	Peso del archivo de instalación de la aplicación, incluyendo el modelo VGG16 transformado TFLite y los cinco módulos Java del Random Forest.
Reducción de Peso TFLite	93.5 %	La Compresión de pesos del modelo VGG16 a TFLite redujo su tamaño de 216 MB (modelo original) a 14 MB optimizando el consumo de almacenamiento y memoria.

Tabla 11: Desempeño y Tamaño de Aplicación en Dispositivo Android

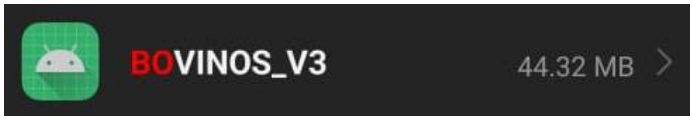


Figura 25: Tamaño que ocupa aplicación en dispositivo Android

9. CONCLUSIONES Y TRABAJOS FUTUROS

9.1. CONCLUSIONES

En respuesta al objetivo general, se desarrolló un sistema integral de predicción de peso bovino que combina visión artificial y aprendizaje automático. Si bien la precisión basada únicamente en imágenes aún no alcanza el nivel de una báscula real, el sistema demuestra ser una alternativa tecnológica viable, accesible y de bajo costo para la estimación visual subjetiva, reduciendo la subjetividad en la comercialización y facilitando procesos en entornos rurales con herramientas prácticas y portables.

Respecto a la preparación del set de datos, se logró conformar y procesar exitosamente un conjunto de datos multimodal compuesto por 513 registros y 17,899 imágenes. La estrategia de selección de imágenes de vista posterior resultó efectiva para capturar la morfología volumétrica del animal. El pre procesamiento, que incluyó la normalización de variables numéricas, la codificación de variables categóricas y la limpieza de registros inconsistentes, garantizó la calidad necesaria para el entrenamiento.

Respecto al entrenamiento de modelos de regresión, se implementaron múltiples arquitecturas, desde modelos clásicos hasta redes neuronales profundas. Los modelos de conjunto (Ensemble Learning), particularmente Random Forest y XGBoost Regressor, demostraron una alta capacidad para capturar relaciones no lineales en datos tabulares. Entre ellos, el XGBoost Regressor se posicionó como el mejor modelo, alcanzando los valores más altos de precisión al presentar $R^2 \approx 0.99$, MAE ≈ 3.27 kg y RMSE ≈ 5.8 kg.

Además, se desarrolló una arquitectura híbrida en serie, integrando una red VGG16 preentrenada (con fine-tuning en el bloque 5) para la extracción de características visuales, combinada con un regresor Random Forest para la predicción final.

Respecto a la evaluación de rendimientos, la evaluación mediante métricas como MAE, RMSE y R^2 permitió contrastar objetivamente los enfoques. Los datos morfométricos tabulares ofrecieron la mayor precisión teórica ($R^2 \approx 0.99$, MAE ≈ 3.27 kg), confirmando su alto valor predictivo.

Por otro lado, el modelo basado exclusivamente en visión artificial alcanzó un desempeño funcional ($R^2 \approx 0.60$, MAE ≈ 29.9 kg), demostrando su utilidad como herramienta de estimación

rápida y sin contacto, aunque con un margen de mejora evidente frente a los métodos tradicionales basados en medidas corporales.

Respecto al desarrollo del prototipo móvil, se materializó la investigación en una aplicación Android funcional que opera bajo el paradigma de Edge Computing. Mediante la optimización de modelos con TensorFlow Lite y la fragmentación del código en Java, se logró que el sistema realice inferencias offline directamente en el dispositivo. Esto valida la utilidad práctica de la herramienta para entornos rurales con conectividad limitada, ofreciendo una interfaz intuitiva que simplifica el proceso tecnológico para el usuario final.

9.2. TRABAJOS FUTUROS

Con base en las limitaciones identificadas en la precisión del modelo de visión artificial y la necesidad de aumentar la robustez comercial del sistema, se proponen las siguientes líneas de trabajo futuro:

- Ampliación y Balanceo del Conjunto de datos para Escalabilidad Comercial: El conjunto de datos actual presenta un sesgo significativo hacia la raza "Local" (68.8%) y el sexo macho (98.8%). Un trabajo futuro crucial consiste en la recolección masiva de datos para aumentar la diversidad fenotípica y racial del inventario y balancear la distribución de sexos y etapas de desarrollo (cría, levante y ceba). Esto garantizará la escalabilidad y la aplicabilidad del sistema en diferentes regiones ganaderas.
- Exploración de Sensores de Profundidad (LiDAR): El peso es una métrica volumétrica. Dado que la estimación bidimensional a partir de una fotografía 2D presenta limitaciones inherentes de precisión, se propone migrar a un enfoque tridimensional. La siguiente etapa investigativa debería explorar la integración de sensores de profundidad, como los escáneres LiDAR. Este volumen estimado se convertiría en un nuevo feature de entrada, con el potencial de elevar la precisión del modelo de visión a niveles comparables con los modelos tabulares.
- Optimización de Arquitecturas Multimodales para la Fusión de Datos: Se sugiere investigar estrategias avanzadas de fusión para la arquitectura multimodal. El intento de fusión paralela en este proyecto demostró ser inestable. Trabajos futuros deben enfocarse en el uso

de mecanismos similares con mejores desarrollo que logren mejorar la convergencia del modelo, explotando las sinergias entre los diferentes tipos de datos.

- Uso de Máscaras y Captura de Vistas Adicionales para Enfoque en Zona Anatómica: Para incrementar la precisión del modelo de visión, se recomienda incorporar técnicas de Segmentación de Instancias (ej. utilizando Mask R-CNN o U-Net) para generar máscaras que aíslen y enfoquen la atención del modelo de regresión exclusivamente en las zonas anatómicas de mayor interés, como el tren posterior, la grupa y el perímetro torácico. Adicionalmente, se debe explorar la captura de vistas adicionales del animal (por ejemplo, perfil lateral o vista dorsal) para que el modelo pueda inferir un mayor número de características morfo métricas que influyen directamente en el peso vivo.

10. REFERENCIAS BIBLIOGRÁFICAS

Bibliografía

[1] C. M. Bishop, Pattern Recognition and Machine Learning., C.M.: Springer, 2006.

[2] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow, California: O'Reilly Media, Inc., 2023.

[3] J. G. W. D. H. T. &. T. R., An Introduction to Statistical Learning, Springer, 2013.

[4] IBM, «IBM,» IBM, 27 Diciembre 2023. [En línea]. Available: https://www.ibm.com/mx-es/think/topics/support-vector-machine. [Último acceso: 14 07 2025].

[5] C. C. a. V. Vapnik, «Support-vector networks,» *SPRINGER NATURE LINK*, vol. 20, nº 3, pp. 273-279, 1995.

[6] N. C. a. J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, London: Cambridge University Press, 2000.

[7] S. A. J. &. S. B., «A tutorial in support Vector regression,» *Statistic and Computing*, vol. 14, nº 3, pp. 199-222, 2004.

[8] S. M. G. R. K. T. a. B. S. K.-R. Müller, «An introduction to kernel-based learning algorithms,» *IEEE Transactions on Neural Networks*, vol. 12, nº 2, pp. 181-201, 2001.

[9] L. Breiman, «Random forests,» *Machine Learning*, vol. 45, nº 1, pp. 5-32, 2001.

[10] A. L. a. M. Wiener, «Classification and regression by randomForest,» *R News*, vol. 2, nº 3, pp. 18-22, 2002.

[11] R. T. a. J. F. T. Hastie, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, New York, NY, US: Springer, 2009.

[12] T. C. a. C. Guestrin, «XGBoost: A scalable tree boosting system,» de *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, 2016.

[13] ". f. a. J. H. Friedman, «A gradient boosting machine,» *Annals of Statistics*, vol. 29, nº 5, pp. 1189-1232, 2001.

[14] S. R. a. R. Gilad-Bachrach, «DART: Dropouts meet multiple additive regression trees,» de *18th Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, San Diego, CA, USA, 2015.

[15] S. Haykin, Neural Networks and Learning Machines, Upper Saddle River, NJ, USA: Pearson, 2009.

[16] Y. B. a. G. H. Y. LeCun, «Deep learning,» *Nature*, vol. 521, pp. 436-444, 2015.

[17] S. GN, «quarkml,» © 2025 · Quark Machine Learning, 6 Abril 2025. [En línea]. Available: https://www.quarkml.com/2023/06/introduction-to-convolutional-neural-networks.html. [Último acceso: 11 11 2025].

[18] S. J. P. a. Q. Yang, «A survey on transfer learning,» *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, nº 10, pp. 1345-1359, 2010.

[19] J. C. Y. B. a. H. L. J. Yosinski, «How transferable are features in deep neural networks?,» de *27th Int. Conf. Neural Information Processing Systems (NIPS)*, Montreal, Canada, 2014.

[20] X. Z. S. R. a. J. S. K. He, «Deep residual learning for image recognition,» de *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016.

[21] S. Mukherjee, «Towards Data Science,» © Insight Media Group, LLC 2025, 18 Agosto 2022. [En línea]. Available: https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758/. [Último acceso: 12 11 2025].

[22] X. Z. S. R. J. S. Kaiming He, «Deep Residual Learning for Image Recognition,» *Microsoft Research,* p. 12, 2015.

[23] K. S. a. A. Zisserman, «Very deep convolutional networks for large-scale image recognition,» *arXiv preprint arXiv*, nº 1409.1556, 2014.

[24] S. Bangar, «Medium,» 28 06 2022. [En línea]. Available: https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f. [Último acceso: 11 11 2025].

[25] A. C. a. P. V. Y. Bengio, «Representation learning: A review and new perspectives,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 35, nº 8, pp. 1798-1828, 2013.

[26] C. A. a. L.-P. M. T. Baltrušaitis, «Multimodal machine learning: A survey and taxonomy,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 41, nº 2, pp. 423-443, 2019.

[27] D. W. a. K. C. A. Kiela, «Deep multimodal learning: A survey on recent advances and trends,» *IEEE Signal Processing Magazine,* vol. 37, nº 6, pp. 82-93, 2020.

[28] A. C. a. P. V. Y. Bengio, «Representation learning: A review and new perspectives,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 35, nº 8, pp. 1798-1828, 2013.

[29] M. Z. H. A. K. T. H. Mobasshir Bhuiya Shagor, «Github,» bhuiyanmobasshir94, 11 August 2022. [En línea]. Available: https://github.com/bhuiyanmobasshir94/CID?tab=readme-ov-file. [Último acceso: 21 Octubre 2024].

[30] FAO, «Guidelines for Body Condition Scoring in Cattle,» ROMA, 2019.

[31] N. –. N. R. Council, «Nutrient Requirements of Beef Cattle,» The National Academies Press, 2021.

[32] R. &. B. K. Rumpf, «Phenotypic assessment of body condition and muscularity in beef cattle using visual scoring,» *Livestock Science,* vol. 212, p. 77–84, 2018.

[33] T. T. R. &. F. J. Hastie, The Element of Statistical Learning., Springer, 2009.

[34] R. S. &. B. A. G. Sutton, Reinforcement Learning: An Instroduction (2nd ed), MIT Press, 2018.

[35] M. I. &. M. T. M. Jordan, «Machine Learning: Trends, perspectives, and prospects.,» *Science,* vol. 349, nº 6245, pp. 255-260, 2015.

[36] A. e. a. (. Barredo Arrieta, «Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI,» *Information Fusion,* vol. 58, pp. 82-115, 2020.

[37] N. R. &. S. H. Draper, Applied Regressions Analysis, Wiley-Interscience, 1998.

[38] O. R. e. al., «ImageNet large scale visual recognition challenge,» *International Journal of Computer Vision,* vol. 115, nº 3, pp. 211-252, 2015.

[39] A. K. M. K. J. N. H. L. a. A. Y. N. H. Ngiam, «Multimodal deep learning,» de *Proc. 28th Int. Conf. Machine Learning (ICML)*, Bellevue, WA, USA, 2011.

ANEXOS

Elemento	DATOS
Anexo A	**Fuente de los Datos Tabulares e Imágenes**
URL	`https://github.com/bhuiyanmobasshir94/CID?tab=readme-ov-file`
Nota	Se incluye un archivo `README.md` detallado con instrucciones de uso.

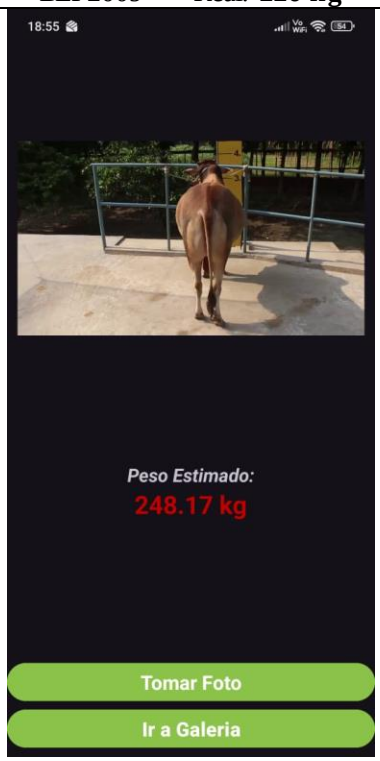
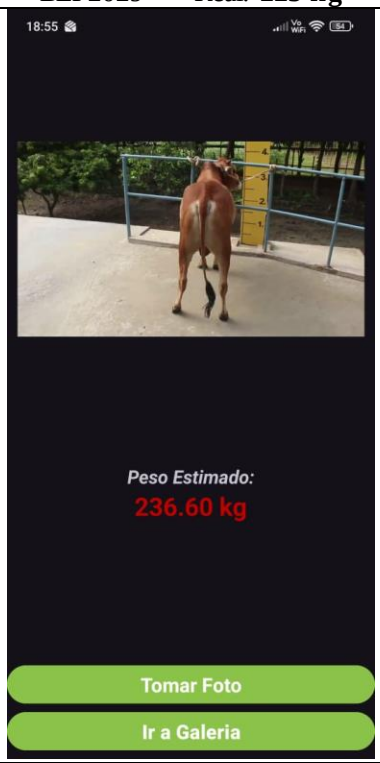
Anexos 1: Fuente Datos.

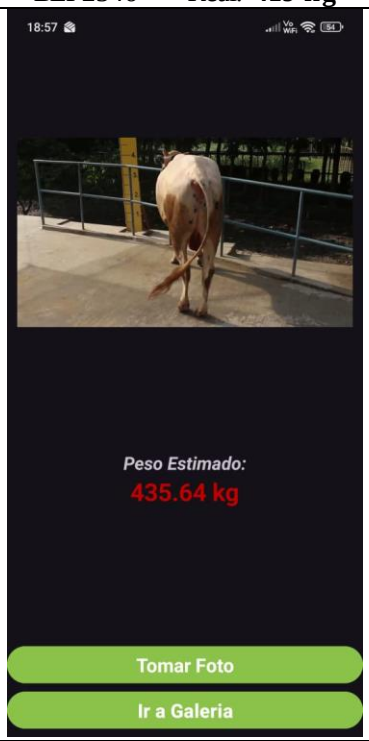
Elemento	Código Modelos Machine Learning y CNN
Anexo B	**Código Fuente de Modelos de ML tradicionales**
URL	`https://github.com/DYNAFEM/REGRESION-PESO-BOVINOS`
Nota	Se incluye un archivo `README.md` detallado con instrucciones del contenido, especificación las ramas de cada modelo así como los datos implementados.

Anexos 2: Repositorio Archivos Modelos ML y CNN

Elemento	Archivo de Códigos APLICACIÓN ANDROID
Anexo A	**Códigos principales transformados para aplicación Android**
URL	`https://github.com/DYNAFEM/APLICACION-ANDROID`
Nota	Se incluye un archivo `README.md` detallado con instrucciones de los archivos disponibles en repositorio.

Anexos 3: Repositorio Archivos Modelos ML y CNN para Android.

BLF2009 => Real: **226 kg**	BLF2029 => Real: **225 kg**	BLF2081 => Real: **300 kg**
		
BLF2183 => Real: **247 kg**	BLF2194 => Real: **196 kg**	BLF2319 => Real: **227 kg**
		

BLF2508 => Real: **315 kg**	BLF2546 => Real: **419 kg**	BLF2606 => Real: **375 kg**
		

Anexos 4: Estimaciones de la Aplicación Android