# Medida de Pawpularity con Deep Learning

Juan Camillo Avendaño Rodriguez
*camilo.avendano1@udea.edu.co*
Ingeniería de sistemas

*Universidad de Antioquia*
Medellín, Colombia

***Contexto-*** Cada año, millones de animales son abandonados o sacrificados en refugios de todo el mundo. Las plataformas de adopción de mascotas juegan un papel crucial para conectar a estos animales con posibles adoptantes, incrementando así sus probabilidades de encontrar un hogar permanente. Sin embargo, la competencia es alta: los refugios y rescatistas tienen que mostrar decenas, a veces cientos, de animales en sus listados, por lo que captar la atención de los usuarios se vuelve un reto.

Las fotografías juegan un papel determinante en este proceso. Estudios y observaciones indican que las mascotas con fotos atractivas tienen más probabilidades de ser adoptadas que aquellas con imágenes de baja calidad[1]. En plataformas como PetFinder.my, las fotos son el principal medio para captar el interés de los adoptantes potenciales. Sin embargo, lo que define una "buena" foto puede variar y no siempre es fácil de cuantificar.

PetFinder.my, la principal plataforma de adopción de animales en Malasia, ha implementado un Cuteness Meter, que analiza fotos de mascotas basándose en ciertos parámetros como la composición de la imagen. Aunque útil, esta herramienta está en una fase experimental y depende de un algoritmo sencillo que tiene mucho margen para mejoras. Aquí es donde el Deep Learning puede aportar un valor significativo, permitiendo desarrollar modelos que analicen de manera más precisa la calidad de las fotos y su "Pawpularity", optimizando la visibilidad de los animales en adopción.

El proyecto que se plantea tiene un impacto real en el bienestar animal, ya que un modelo eficaz puede aumentar la tasa de adopción y, como consecuencia, mejorar la vida de miles de animales. Este contexto también es relevante para demostrar habilidades avanzadas en Ciencia de Datos y Deep Learning, ya que combina el análisis de imágenes y datos tabulares para abordar un problema de clasificación y regresión en un entorno de la vida real.

## I. Introducción

El objetivo principal de este proyecto es desarrollar un modelo de **Machine Learning** que sea capaz de predecir la "Pawpularity" de una imagen de mascota. Este puntaje refleja el atractivo de la foto y está correlacionado con el interés que genera entre los usuarios, medido a través de métricas como las visitas a la página y las interacciones con el perfil del animal.

En términos técnicos, este proyecto tiene dos componentes principales:

1. **Predicción basada en imágenes**: Utilizando un modelo de redes neuronales convolucionales (**CNN**), entrenaremos el sistema para analizar la composición y calidad visual de las fotos. Las redes CNN son ideales para este tipo de problemas porque pueden captar patrones visuales en las imágenes (como enfoque, iluminación, etc.) que influyen en la percepción del atractivo, así extraer estas características para llegar a un modelo objetivo de regresión basada en metadatos.
2. **Predicción basada en metadatos**: el modelo también considerará un conjunto de características tabulares (metadatos) que incluyen información clave sobre la calidad visual, presencia de elementos adicionales (como humanos o accesorios), y otras características, el cual apunta a sacar la medida de pawpularity. Este enfoque híbrido permitirá al modelo aprender tanto de la imagen como de los descriptores manuales.

El objetivo es no solo predecir con precisión el puntaje de Pawpularity, sino también interpretar el impacto de ciertos atributos en la calidad de las fotos. A nivel de negocio, un modelo exitoso permitirá hacer recomendaciones automáticas sobre cómo mejorar la calidad de las fotos, por ejemplo, sugiriendo ajustes en el ángulo, iluminación o presencia de accesorios.

Además, este reto es parte de un curso donde se espera demostrar conocimientos en Deep Learning. El objetivo es no solo entrenar un modelo predictivo, sino también aplicar técnicas de preprocesamiento de imágenes, análisis de datos y evaluación de modelos, utilizando métricas relevantes como el **Root Mean Squared Error (RMSE)[2]** para medir el rendimiento del modelo.

## II. Dataset

- **Tipo de datos**:
    - Imágenes de mascotas en formato .jpg.

    - Metadatos en formato tabular, que incluyen características como el enfoque, la presencia de humanos, accesorios, y más.
- **Tamaño**:
    - Aproximadamente **9,912** imágenes de entrenamiento.
    - El archivo **`train.csv`** contiene metadatos sobre cada foto y su respectivo puntaje de "Pawpularity".
    - Tamaño en disco: Aproximadamente **1 GB** (para imágenes y metadatos combinados).
- **Distribución de las clases**:
    - El objetivo es predecir un puntaje de "Pawpularity", que es una variable continua (no categórica). La distribución del puntaje varía entre 0 y 100 la cual tiene la siguiente distribución

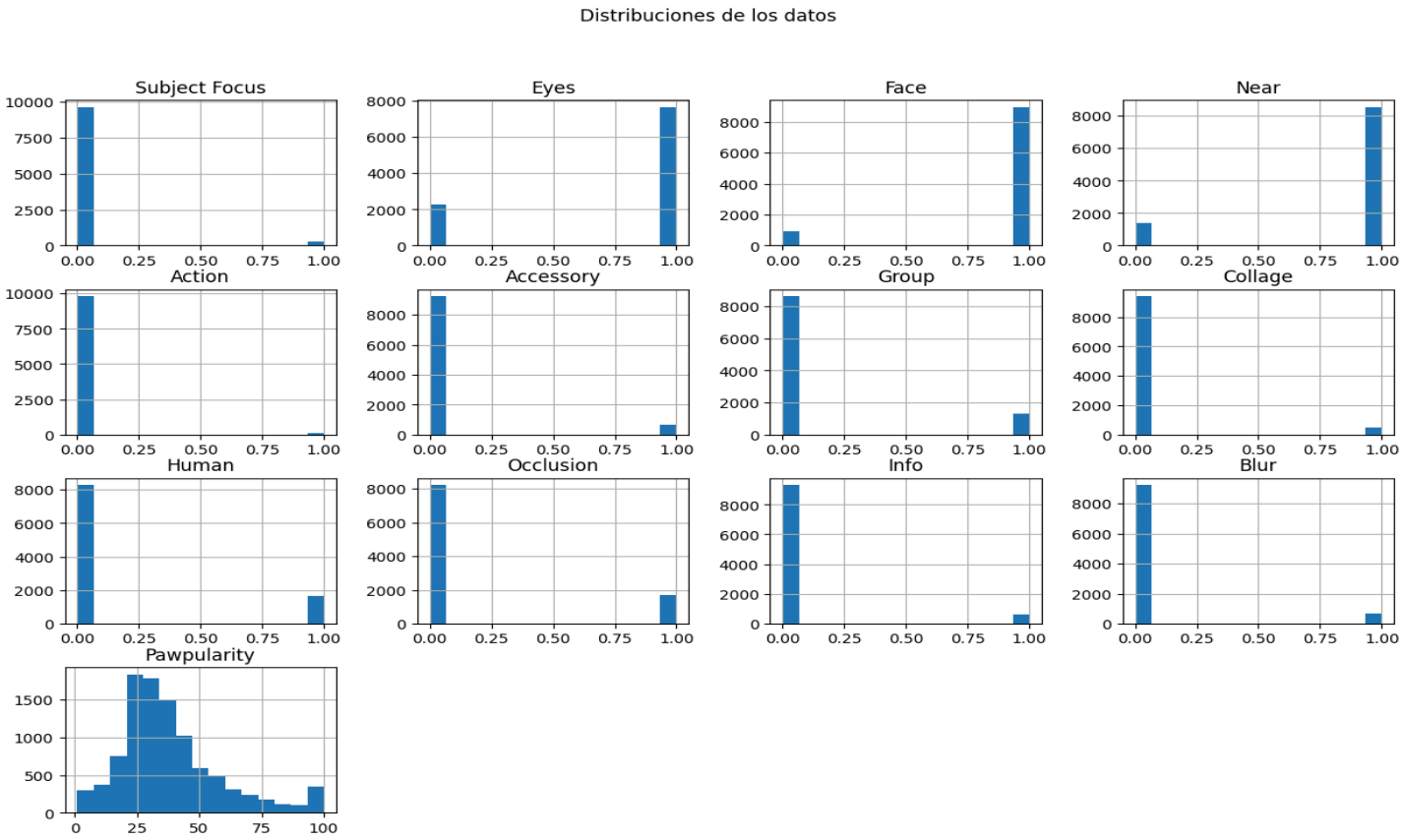


Figura 1

Distribución de los metadatos de las imágenes

## III. Métricas de desempeño

La métrica principal de evaluación será el **Root Mean Squared Error (RMSE)**, ya que estamos tratando con un problema de regresión. RMSE mide la diferencia entre los valores reales y los valores predichos por el modelo, y minimizaremos esta métrica para optimizar el desempeño del modelo.

- **Métricas de Negocio**:

A largo plazo, el éxito del modelo se medirá por su capacidad de mejorar la tasa de adopción de mascotas, reflejada en la reducción del tiempo promedio que una mascota permanece en adopción. Un modelo efectivo ayudaría a optimizar los perfiles de las mascotas, haciéndolas más atractivas para los posibles adoptantes.

## IV. ESTRUCTURA DE LOS NOTEBOOKS

El trabajo se organizó en un flujo coherente que permitió la construcción, evaluación y mejora del modelo. Los notebooks utilizados contienen las siguientes secciones:

1. Exploración de Datos:
   - Análisis inicial de los datos disponibles en los archivos CSV.
   - Visualización de las características de entrada, incluyendo correlaciones y distribuciones de variables.
2. Preprocesamiento de Imágenes y Datos:
   - Normalización de imágenes a un tamaño estándar (128x128 píxeles).
   - Generación de tensores a partir de las rutas de las imágenes.
   - Codificación one-hot para tareas de clasificación.
3. Desarrollo del Modelo:
   - Creación de dos modelos: uno basado en regresión directa para predecir Pawpularity y otro utilizando clasificación con categorías predefinidas.
   - Uso de redes convolucionales (CNN) para extraer características de las imágenes.
4. Entrenamiento y Evaluación:
   - Configuración de los hiperparámetros del modelo y generación de datos aumentados.
   - Evaluación del desempeño del modelo utilizando métricas como RMSE, MAE y MAPE.
5. Iteraciones y Mejoras:
   - Ajuste de arquitecturas, capas y funciones de activación.
   - Exploración de diferentes estrategias para predecir Pawpularity.

## V. EXPLORACIÓN DE DATOS

Se llevó a cabo una exploración inicial de los datos para comprender mejor las características y su relación entre sí. Todas las variables presentes en el conjunto de datos son de tipo binario, lo que implica que cada característica toma solo dos posibles valores (por ejemplo, 0 y 1). Para evaluar la relevancia de cada variable, se analizó la matriz de correlación, la cual mostró que las características no presentan una alta correlación entre sí. Esto sugiere que cada variable aporta información única al modelo, lo que es positivo para evitar redundancias y mejorar la capacidad predictiva del modelo.

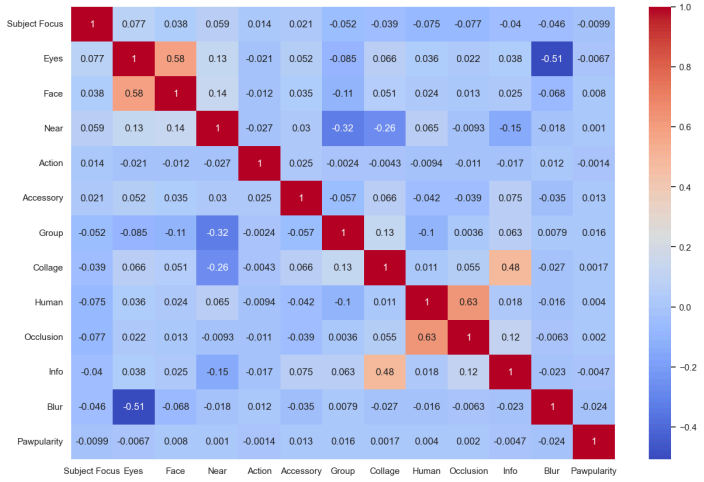


Figura 2.
Matriz de Correlación

Se realizó una impresión en pantalla de los datos de las imágenes con el objetivo de explorar sus características clave. Esto incluyó la visualización de las dimensiones de cada imagen, el identificador único (ID) asociado a cada una, y la variable "Pawpularity", que representa la medida de popularidad relacionada con cada imagen.

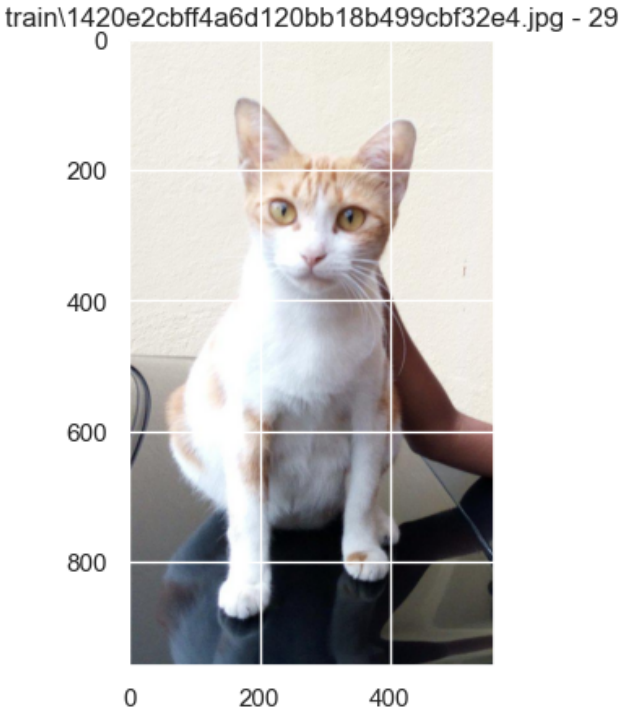


Figura 3.
Ejemplo de dataset y su “Pawpularity”.

Además, se desarrolló una función que permite imprimir en pantalla cinco imágenes de diferentes IDs, todas asociadas a una calificación específica. En este caso, las siguientes imágenes corresponden a una "Pawpularity" de 20. Esta función facilita la visualización de cómo las imágenes con la misma calificación se distribuyen en el conjunto de datos, proporcionando una manera rápida de inspeccionar ejemplos representativos de cada categoría.

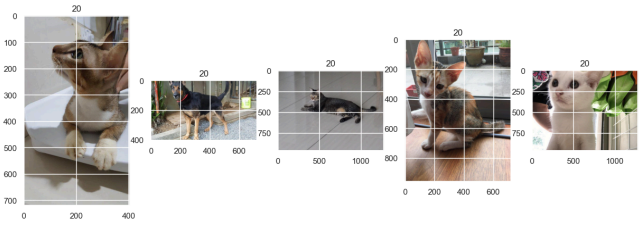

Figura 4.

Conjunto de 5 imágenes con 20 “Pawpularity”.

El siguiente conjunto de imágenes corresponde a una “Pawpularity” de 100.

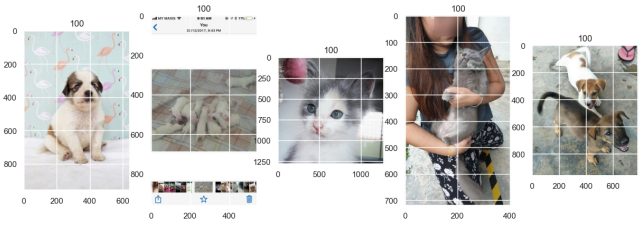

Figura 5.

Conjunto de 5 imágenes con 20 “Pawpularity”.

Se realizó una división de las imágenes en cuatro categorías según su nivel de "Pawpularity", utilizando los siguientes rangos: 0-25, 25-50, 50-75 y 75-100. Al analizar la distribución, se observa que la mayoría de las imágenes se agrupan en la categoría "D", que corresponde al rango más bajo (0-25). Esto indica que una gran parte de las imágenes en el conjunto de datos tienen puntuaciones de popularidad relativamente bajas, lo que podría sugerir que el conjunto de datos está sesgado hacia imágenes con menor atractivo o visibilidad.

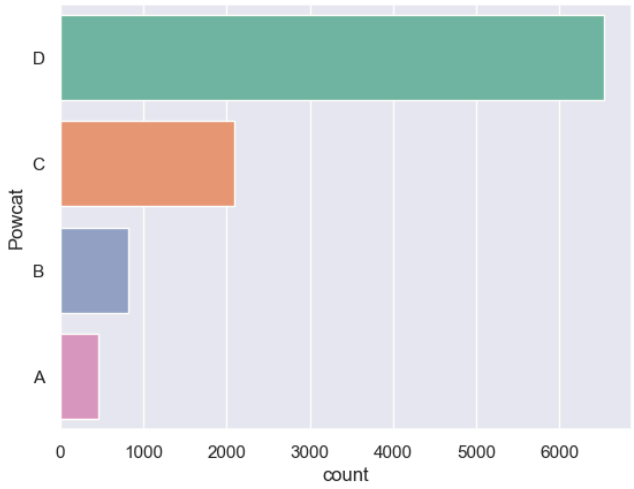


Figura 6.

Distribución de datos en “Pawpularity” en categorías.

Lo cual se confirma al observar la distribución de los valores de "Pawpularity" en el conjunto de datos. La mayor concentración de imágenes en el rango más bajo sugiere que una gran parte de las imágenes no alcanzan niveles altos de popularidad, lo que refuerza la hipótesis de que la mayoría de las imágenes en el conjunto de datos son menos populares.

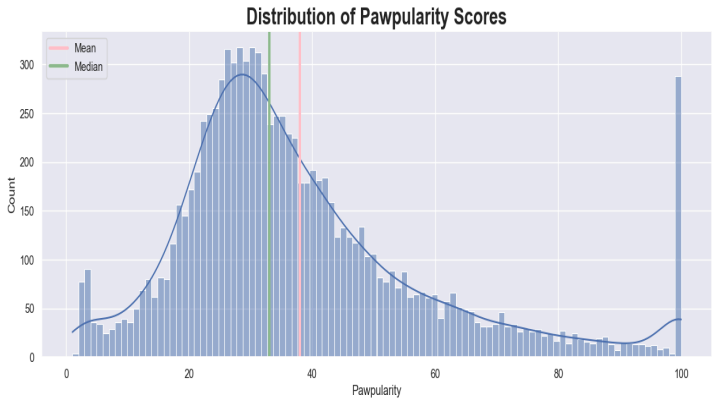


Figura 7.

Distribución de datos en “Pawpularity”.

Se desarrolló una función que permite visualizar tres tipos de gráficos para analizar el impacto de una variable en la puntuación de "Pawpularity". A continuación, se presenta un ejemplo que muestra cómo la variable "Eyes" influye en la distribución de la puntuación de "Pawpularity". Los gráficos generados facilitan la comprensión de cómo la presencia o ausencia de ojos en las imágenes se asocia con diferentes niveles de popularidad.

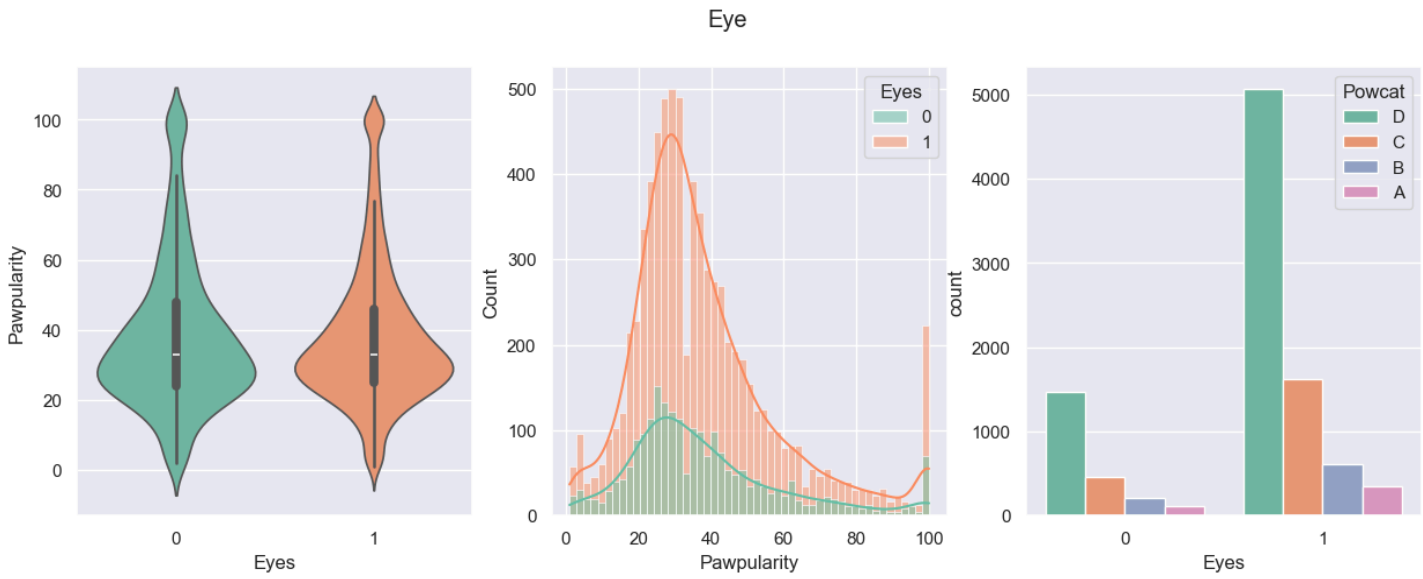


Figura 8.

Impacto de la variable “Eye” en su “Pawpularity”.

## VI. ENTRENAMIENTO

Para asegurar un uso eficiente de los recursos del sistema durante el entrenamiento del modelo, se configuró TensorFlow para aprovechar la GPU disponible. En este caso, una **NVIDIA GeForce RTX 4060**. Esto es fundamental para acelerar el entrenamiento del modelo, ya que las GPUs son mucho más rápidas que las CPUs en operaciones de cómputo intensivo, como las requeridas en redes neuronales.

Adicionalmente se configuró **el Crecimiento Dinámico de Memoria en la GPU:** Para evitar que TensorFlow asigne toda la memoria de la GPU de forma inmediata, se habilitó el crecimiento dinámico de memoria. Esto permite que la memoria de la GPU se asigne de manera progresiva según sea necesario durante el entrenamiento, evitando así posibles problemas de agotamiento de memoria si el modelo se vuelve más grande o si se entrenan lotes de datos más grandes.

**Preprocesamiento de datos:**

- **Generación de rutas a imágenes:** Se definieron funciones para generar rutas a los archivos de imagen utilizando sus IDs únicos, facilitando el acceso a los datos para el entrenamiento y prueba.
- **Normalización y redimensionado de imágenes:** Las imágenes se preprocesaron para:
    - Escalar los valores de los píxeles a un rango entre 0 y 1.
    - Redimensionarlas a 128x128 píxeles para uniformizar la entrada al modelo.
- **Conversión a arrays NumPy:** Las imágenes preprocesadas se almacenaron como tensores en un array NumPy, lo que permite una manipulación más eficiente durante el entrenamiento.
- **División de datos:** El conjunto de datos se dividió en un conjunto de entrenamiento y un conjunto de prueba (80%-20%), asegurando una evaluación adecuada del modelo.

**Arquitectura del modelo:**
El modelo inicial está diseñado para observar el comportamiento de los datos y establecer una línea base para futuras iteraciones. La arquitectura incluye los siguientes componentes principales:
**Capa de Entrada:**

- Recibe imágenes con dimensiones (128, 128, 3) para representar imágenes RGB redimensionadas.

**Capas Convolucionales:**

- Se utilizan múltiples capas convolucionales para extraer características de las imágenes, comenzando con filtros más simples y aumentando gradualmente la complejidad:
    - **Capa 1:** 16 filtros, tamaño de kernel (7, 7), stride (2, 2).
    - **Capa 2:** 32 filtros, tamaño de kernel (3, 3), stride (2, 2).
    - **Capa 3 y siguientes:** 64 y 128 filtros para extraer características más detalladas.

**Normalización por Lotes (Batch Normalization):**

- Después de varias capas convolucionales, se aplica batch normalization para estabilizar y acelerar el entrenamiento al normalizar las salidas de las capas.

**Capa de Max Pooling:**

- Reduce la dimensionalidad espacial de las características aprendidas para manejar mejor la complejidad del modelo.

**Dropout:**

- Se emplean capas de dropout con tasas de 0.25 y 0.5 para prevenir el sobreajuste al ignorar de forma aleatoria algunas neuronas durante el entrenamiento.

**Capa densa final:**

- Una capa completamente conectada (Dense) de 512 unidades seguida de la salida final de una unidad para predecir el puntaje continuo de "Pawpularity".

El modelo se compila con:

- **Optimizador:** Adam, para manejar eficientemente la actualización de pesos.
- **Función de pérdida:** Error cuadrático medio **(mse)** para ajustarse al problema de regresión.
- **Métricas de evaluación:** RMSE, MAE y MAPE para medir el desempeño

**Aumentación de Datos**

- Se implementa ImageDataGenerator para realizar aumentación de datos en tiempo real, aumentando la diversidad del conjunto de entrenamiento mediante transformaciones como:
    - Rotación (15 grados).
    - Zoom (15%).
    - Desplazamientos horizontales y verticales.
    - Inversiones horizontales.
    - Modificaciones de corte y transformación.

## Entrenamiento del Modelo

- El modelo se entrenó durante **3 épocas iniciales** utilizando datos aumentados. Se utilizó un tamaño de lote de 32 imágenes y el conjunto de prueba se mantuvo constante para la validación.

## Resultados del Entrenamiento inicial:

- RMSE: Entrenamiento (21.78) y Validación (21.29) indican errores significativos en la escala de 0 a 100.
- MAE: Entrenamiento (16.29) y Validación (15.14) muestran un error promedio elevado.
- MAPE: Ambos conjuntos reflejan dificultades para ajustar valores pequeños, aunque mejora en validación

**Optimización de Hiper Parámetros:**

Para mejorar el desempeño del modelo inicial, se utilizó un enfoque de búsqueda de hiperparámetros dinámico mediante la biblioteca Keras Tuner. Esto permitió explorar diversas configuraciones de la arquitectura del modelo y los hiperparámetros clave para identificar la mejor combinación para predecir la "Pawpularity".

**Construcción del Modelo con Hiperparámetros Dinámicos:**

- La función build_model(hp) definió una arquitectura adaptable a valores dinámicos para:
    - Tamaño de los filtros en capas convolucionales (filters_layer_1, filters_layer_2, filters_layer_3).
    - Tamaño del kernel en convoluciones (kernel_size).
    - Número de unidades densas en la capa totalmente conectada (dense_units).
    - Tasa de dropout para regularización (dropout_rate).
    - Tasa de aprendizaje (learning_rate).
- Este diseño flexible permitió explorar múltiples combinaciones de hiperparámetros durante la optimización.

**Configuración del Proceso de Búsqueda:**

- **Método:** Se utilizó el algoritmo **Hyperband**, que realiza una búsqueda eficiente mediante el ajuste de recursos computacionales asignados a cada combinación.
- **Objetivo:** Minimizar la métrica de validación val_rmse.
- **Parámetros de búsqueda:**
    - Máximo de épocas: 10.
    - Factor de reducción: 3.

**Ejecución del Proceso:**

- La búsqueda iterativa evaluó diferentes combinaciones de hiperparámetros, seleccionando aquellas con el menor val_rmse.
- **Resultados de las Pruebas:**
    - Total de pruebas: 30.
    - Mejor val_rmse: **20.93** (mejor que el modelo inicial).

**Mejor Configuración Encontrada:**

- **Hiperparámetros óptimos:**
    - filters_layer_1: 32.
    - filters_layer_2: 256.
    - filters_layer_3: 128.

    - dense_units: 128.
    - dropout_rate: 0.3.
    - learning_rate: 0.0001.
- Épocas finales del mejor modelo: 4.

**Progreso Inicial:** El modelo muestra un aprendizaje efectivo durante las primeras etapas, como lo indica la reducción significativa de las métricas de pérdida (loss), error absoluto promedio (mae), y error cuadrático medio (rmse) entre las épocas 1 y 2. Esto refleja que el modelo comienza a ajustarse a los datos de entrenamiento.

**Estabilidad:** A partir de la segunda época, las métricas de validación (val_loss y val_rmse) se estabilizan, sugiriendo que el modelo ha alcanzado un punto de convergencia. Sin embargo, este punto no es óptimo, ya que las métricas aún reflejan un error considerable.

**Error Relativo Alto**: A pesar del progreso, las métricas de error (rmse y mape) indican que el modelo no predice con alta precisión. Un rmse cercano a 21 sigue siendo significativo en comparación con la escala de "Pawpularity", que varía de 0 a 100. Esto sugiere la necesidad de mejoras adicionales en la arquitectura o los hiperparámetros.

**Posible Sobreajuste:** El incremento en las métricas de validación (val_loss y val_rmse) en la tercera época podría ser un signo temprano de sobreajuste, indicando que el modelo empieza a ajustar demasiado a los datos de entrenamiento en lugar de generalizar bien para los datos de prueba.

**Segundo enfoque del Modelo:**

En este enfoque, se reestructuró el problema de regresión a un problema de clasificación. Se dividió la puntuación de "Pawpularity" en 4 categorías, transformando las etiquetas en vectores one-hot codificados. Este cambio permite al modelo predecir la probabilidad de cada imagen perteneciendo a una categoría específica.

**Preparación de los Datos**

1. **Conversión a One-Hot Encoding:**
    - Las puntuaciones originales de "Pawpularity" se transformaron en una representación categórica con 4 clases.
    - Se implementó la función convert2onehot_4 para convertir las etiquetas en vectores one-hot.
    - Ejemplo de transformación:
        - Puntuación 0 → [1, 0, 0, 0]
        - Puntuación 1 → [0, 1, 0, 0]
        - Puntuación 2 → [0, 0, 1, 0]
        - Puntuación 3 → [0, 0, 0, 1]
2. **Formato de los Datos:**
    - **Entrenamiento:** (7929, 128, 128, 3) para las imágenes y (7929, 4) para las etiquetas.
    - **Prueba:** (1983, 128, 128, 3) para las imágenes y (1983, 4) para las etiquetas.

**Arquitectura del Modelo**

1. **Capas Convolucionales:**
    - Similar al modelo inicial, con capas convolucionales y de batch normalization para extraer características de las imágenes.
    - Se emplearon filtros crecientes (16, 32, 64, 128) y técnicas de pooling para reducir la dimensionalidad.
2. **Capas Completamente Conectadas:**
    - Una capa densa de 512 neuronas, seguida de Dropout para regularización.
    - La capa de salida tiene 4 neuronas (una por categoría), que generan probabilidades para cada clase.
3. **Compilación:**
    - **Función de pérdida:** Error cuadrático medio (mse).
    - **Métricas:** RMSE, MAE, y MAPE para evaluar el desempeño.

**Conclusiones del Segundo Enfoque**

1. **Desempeño Mejorado:**
    - Los valores de val_loss y val_rmse muestran una reducción significativa con respecto al primer enfoque, indicando que la clasificación por categorías puede capturar mejor las características de los datos.
    - Un val_rmse final de 0.2135 sugiere que el modelo puede predecir con mayor precisión la categoría de "Pawpularity".
2. **Ventajas del Enfoque:**
    - Simplificación del problema al convertir una tarea de regresión en una de clasificación.
    - Mejor ajuste en datos donde las diferencias entre puntuaciones son difíciles de capturar.

## VII. Conclusión

El trabajo realizado demuestra un progreso significativo en el desarrollo de un modelo capaz de predecir la “Pawpularity” de las imágenes de mascotas. El cambio de enfoque de regresión a clasificación fue clave para mejorar el desempeño, y las herramientas utilizadas, como la optimización de hiperparámetros, mostraron su utilidad. Este proyecto no solo presenta un avance técnico, sino que también tiene un impacto potencial real en aumentar las tasas de adopción y mejorar el bienestar animal.

Este informe refleja un proceso completo de aprendizaje, experimentación y mejora continua, sentando una base sólida para futuros trabajos en esta área.

## VIII. REFERENCIAS

[1] "Efecto de superioridad de la imagen: ¿qué es y cómo nos afecta?" https://psicologiaymente.com/psicologia/efecto-superioridad-imagen(accessed Sept 7, 2024).

[2] RMSE, https://docs.oracle.com/cloud/help/es/pbcs_common/PFUSU/insights_metrics_RMSE.htm#PFUSU-GUID-FD9381A1-81E1-4F6D-8EC4-82A6CE2A6E74 (accessed Sept 7, 2024)

[3] Challenge Kaggle, “PetFinder.my - Pawpularity Contest”, https://www.kaggle.com/competitions/petfinder-pawpularity-score/overview (accessed Aug 24, 2024 )