

## PRESENTACION DE PROYECTO INTEGRADOR

# VISION ARTIFICIAL DE APILAMIENTO DE PANALES – ANALISIS EDA

El EDA en visión por computadora no se centra en medias y desviaciones estándar de columnas, sino en **propiedades visuales de las imágenes** (tamaño, color, calidad, balance de clases, embeddings).

Profesor	Prof. Gladys Villegas, PhD.
Materia	PROYECTO INTEGRADOR EN INTELIGENCIA ARTIFICIAL – MIAR0545
Alumnos	Francisco Javier Estupiñan Andrade David Alejandro Narváez Mejía
Fecha	09/24/2025

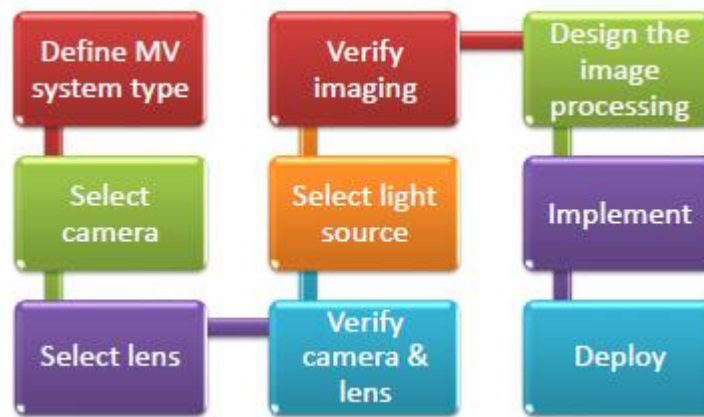
Online  
Universidad  
Espíritu Santo

# Introducción a la Inteligencia Artificial

## TABLA DE CONTENIDO

1.	FLUJOGRAMA DE PROCESO – VISION POR COMPUTADOR .....	3
2.	EDA EN VISION POR COMPUTADOR .....	5
2.1	CARGA Y EXPLORACIÓN INICIAL .....	5
2.2	ANÁLISIS UNIVARIADO (NIVEL IMAGEN) .....	5
2.3	ANÁLISIS BIVARIADO .....	5
2.4	ANÁLISIS MULTIVARIADO .....	5
2.5	DETECCIÓN DE ANOMALÍAS .....	5
2.6	CONCLUSIONES E INSIGHTS .....	5
2.7	TABLA EDA DATO VS IMAGEN .....	5
3.	NOTEBOOK EDA DE VISION POR COMPUTADOR .....	7
3.1	INTRODUCCIÓN .....	7
3.2	PREPARACIÓN DEL ENTORNO .....	7
3.3	CREACIÓN DEL DATASET .....	7
3.4	VISUALIZACIÓN INICIAL .....	7
3.5	IMPORTACIÓN DE LIBRERÍAS Y ANÁLISIS .....	7
3.6	TABLA COMPARATIVA DE MODOS .....	8
4.	ANÁLISIS DE RESULTADOS .....	9
4.1	EXPLORACIÓN INICIAL .....	9
4.2	DISTRIBUCIÓN DE CLASES .....	9
4.3	CORRELACIONES Y PATRONES .....	9
4.4	DISTRIBUCIONES ANÓMALAS E INCONSISTENCIAS .....	9
4.5	INSIGHTS PRINCIPALES .....	10
4.6	CONCLUSIONES .....	10
5.	REFERENCIAS .....	12

## 1. FLUJOGRAMA DE PROCESO – VISION POR COMPUTADOR



*Figura 1: Proceso de proyectos Visión por Computador / FUENTE: Machine-Vision-Systems-Design A3 VISION*

El flujo de proceso para el diseño e implementación de un proyecto de visión por computador, tiene la siguiente **descripción paso a paso** que acompañará al diagrama “Typical Design Sequence”:

1. Definir el tipo de sistema de visión (MV)
  - Propósito: elegir la arquitectura general (smart camera, cámara atada a controlador, PC-based).
  - Entradas: objetivos de calidad, tasa de producción, espacio disponible, presupuesto, integración con PLC/robot.
  - Salidas: arquitectura elegida, cantidad de cámaras, esquema de disparo (trigger) y sincronización.
  - Nota proyecto: si hay varias vistas o alta complejidad (mediciones y ML), suele convenir **PC-based** por flexibilidad.
2. Seleccionar la(s) cámara(s)
  - Definir **campo de visión (FOV)** con margen (10–20% de overscan).
  - Calcular **resolución espacial** requerida (mm/px) a partir del tamaño mínimo a medir y los “píxeles que deben abarcar la característica”.
  - Validar **velocidad** (fps y tiempo de exposición para que el desenfoque por movimiento  $\leq 1$  px), **obturador** (global preferible), **interfaz** (GigE/USB3/CoaXPress).
  - Salidas: modelo de cámara, resolución, fps, obturador, interfaz.
  - Nota proyecto: si la línea se mueve rápido, priorizar **global shutter** y exposición corta.
3. Seleccionar el lente
  - Calcular magnificación (sensor/FOV) y distancia de trabajo.
  - Elegir focal que cumpla el FOV en la distancia real y resolución óptica  $\geq$  a la frecuencia de Nyquist del pixel.
  - Verificar montura (C/CS/S) y distorsión aceptable; considerar telecéntrico si la precisión dimensional es crítica.
  - Salidas: focal, distancia de trabajo, montura, requisitos de resolución óptica.
4. Verificar cámara & lente
  - Prueba en banco: enfocar al FOV objetivo, medir **distancia de trabajo**, chequear **resolución** en centro y esquinas con diana, revisar **distorsión**.
  - Criterio de salida: se cumplen FOV, nitidez y resolución en todo el campo.
5. Seleccionar la técnica/fuente de iluminación
  - Partir de la **característica que necesita contraste** (bordes de panales, alineación, huecos, etc.).
  - Elegir **geometría** (backlight para siluetas, front light difusa/oscura/brillante), **difusividad**, **longitud de onda**, **polarización**, y si será **continua o estroboscópica**.
  - Salida: modelo y geometría de iluminación + fijación mecánica.

- Nota proyecto: para contorno y gaps, un **backlight difuso** suele dar siluetas limpias; si debes ver textura/superficie, front-light difusa.
6. Verificar la imagen (imaging)
    - Montaje preliminar: capturar lote de muestras representativas.
    - Revisar **exposición, uniformidad, contraste, reflejos, sombras y SNR**; ajustar ángulos/altura/ganancia.
    - **Integrar EDA de imágenes** aquí: histogramas de brillo, blur (var. del Laplaciano), distribución de tamaños, balance de clases, duplicados (pHash), outliers.
    - Salidas: parámetros finales de cámara/iluminación y **plan de normalización** (resize, recorte, normalización RGB) + **plan de augmentation** según variabilidad real.
  7. Diseñar el procesamiento de imagen
    - Definir **pipeline** a partir del EDA y requisitos:
      1. Preprocesado (resize/padding, normalización, reducción de ruido).
      2. Localización/segmentación (bordes, umbrales adaptativos, morfología, o modelo CNN/Detección).
      3. Cálculos/medidas (alineación, separaciones, conteo, reglas geométricas).
      4. Clasificación OK/Defecto o métricas de calidad.
    - Elegir herramientas (clásicas vs. **CNN/CNN+Transformers**) en función de la separabilidad observada en el EDA (embeddings/PCA).
    - Definir **KPIs de desempeño** (precisión, recall, F1, tiempo de ciclo) y protocolo de pruebas (FAT).
    - Salidas: diagrama de bloques, parámetros iniciales, dataset curado y dividido (train/val/test).
  8. Implementar
    - Integración con **PLC/robot/HMI**, triggers, protocolos (EtherNet/IP, Profinet, etc.).
    - Calibraciones requeridas (intrínseca; mano-ojo si hay robot y medidas en mm).
    - Construcción mecánica, cableado, seguridad, receta de producto.
    - Pruebas funcionales: **FAT** (en taller) con criterios aceptados (tasas de falsos  $\pm$  máximas, tiempos de ciclo).
  9. Desplegar
    - Instalación, **SAT** (en planta), validación en producción, capacitación a operadores/mantenimiento.
    - Entrega de **documentación** (parámetros, planos, backups, manuales) y **plan de soporte**.
    - Estrategia de **monitoreo y mejora continua** (log de imágenes, re-entrenos, umbrales adaptativos, mantenimiento de iluminación/cámara).

## 2. EDA EN VISION POR COMPUTADOR

El EDA en visión por computadora no se centra en medias y desviaciones estándar de columnas, sino en **propiedades visuales de las imágenes** (tamaño, color, calidad, balance de clases, embeddings).

### 2.1 CARGA Y EXPLORACIÓN INICIAL

Revisar estructura del dataset:

- ✓ Número total de imágenes.
- ✓ Número de clases (si es clasificación).
- ✓ Cantidad de imágenes por clase (balance/desbalance).

Ver ejemplos de imágenes de cada clase para validar calidad y coherencia.

Revisar tamaños y formatos (JPG, PNG, resoluciones).

### 2.2 ANÁLISIS UNIVARIADO (NIVEL IMAGEN)

**Dimensiones:** histogramas de ancho, alto y proporción (aspect ratio).

**Distribución de colores:** histograma de valores RGB o escala de grises.

**Tamaño de archivo:** ver si hay mucha variación.

**Calidad:** detectar imágenes borrosas, muy oscuras o sobreexpuestas.

### 2.3 ANÁLISIS BIVARIADO

Comparar características de las imágenes con sus etiquetas:

- ✓ Número de imágenes por clase → detectar desbalance.
- ✓ Relación entre tamaño de la imagen y la clase.
- ✓ Variación de color promedio por clase (ejemplo: panales vs jarras podrían tener gamas distintas).

### 2.4 ANÁLISIS MULTIVARIADO

Reducción de dimensionalidad:

- ✓ Extraer embeddings con un modelo preentrenado (ejemplo: ResNet, MobileNet).
- ✓ Usar PCA o t-SNE para visualizar cómo se agrupan las clases en 2D/3D.

Detectar si las clases están bien separadas visualmente o si hay mucho solapamiento.

### 2.5 DETECCIÓN DE ANOMALÍAS

Imágenes duplicadas.

Imágenes mal etiquetadas (ej. un panal en la clase “barril”).

Imágenes vacías, corruptas o con ruido.

Variaciones extremas en iluminación, fondo o perspectiva.

### 2.6 CONCLUSIONES E INSIGHTS

Identificar si se necesita:

- ✓ Balancear clases (data augmentation o recolección de más datos).
- ✓ Normalizar tamaños (redimensionar todas a 224x224, por ejemplo).
- ✓ Mejorar calidad (filtrar imágenes borrosas).
- ✓ Preprocesamiento de color (normalización de canales RGB).

### 2.7 TABLA EDA DATO VS IMAGEN

En datos tabulares, el EDA se centra en estadísticas, correlaciones y valores faltantes.

En imágenes, el EDA se centra en calidad visual, balance de clases, propiedades de color/forma y en detectar anomalías visuales.

El punto común es que ambos buscan preparar los datos para un modelo de IA robusto.


Aspecto	EDA en Datos Tabulares (numéricos/catégoricos)	EDA en Imágenes (visión por computador)	Qué se contrasta
Estructura del dataset	Número de filas y columnas, tipos de variables (int, float, catégoricas, fechas)	Número total de imágenes, número de clases, formato (JPG, PNG, TIFF), dimensiones (alto × ancho)	Tamaño y consistencia del dataset
Valores faltantes / corruptos	Detección de <b>NaN</b> , duplicados, registros incompletos	Imágenes corruptas, vacías, duplicadas (hash), ilegibles	Calidad y completitud de la información
Balance de clases	Conteo de catégorías (ej. hombres vs mujeres, aprobado vs reprobado)	Número de imágenes por clase (OK vs Defecto), desbalance de clases	Distribución equitativa de etiquetas
Estadísticas univariadas	Media, mediana, moda, desviación estándar, rangos	Histograma de brillo, color (RGB, HSV), tamaños de imagen, blur (nitidez)	Distribución de valores/características
Outliers	Detección por IQR, Z-score, clustering	Imágenes atípicas (muy oscuras, sobreexpuestas, con ruido, mal encuadre)	Consistencia y representatividad
Correlaciones	Matriz de correlación (Pearson, Spearman) entre variables	Relación entre atributos visuales y clases (ej. color medio vs clase)	Variables relevantes para predicción
Multivariable / Reducción de dimensionalidad	PCA, t-SNE, clustering de variables	PCA/t-SNE sobre embeddings (CNN preentrenada) para ver agrupamiento de imágenes por clase	Separabilidad entre clases
Temporalidad (si aplica)	Series de tiempo, estacionalidad, tendencias	Secuencias de video, patrones de movimiento	Análisis de patrones en el tiempo
Conclusiones	Identificar variables clave, preparar limpieza y normalización 	Identificar preprocesamiento: resize, normalización de color, augmentations necesarias	Recomendaciones para el modelo de IA

Tabla 1: Comparativa de dato vs imagen / FUENTE: Generado ChatGPT

### 3. NOTEBOOK EDA DE VISION POR COMPUTADOR

Este notebook busca enseñar paso a paso cómo realizar un **Análisis Exploratorio de Datos (EDA)** para datasets de **visión por computador**, integrando diferentes modos (clasificación, detección y segmentación).

Los ítems son:

1. **Organizar datasets** de forma estandarizada.
2. **Visualizar imágenes y etiquetas** para detectar errores o anomalías.
3. **Aplicar métricas descriptivas** que orienten la fase de preprocesamiento y modelado.

#### 3.1 INTRODUCCIÓN

El notebook se titula EDA de Visión por Computador — Multifomato y corresponde al proyecto integrador de la maestría.

Explica que el análisis exploratorio de datos (EDA) soporta tres tipos de tareas en visión por computador:

**Clasificación:** imágenes organizadas en carpetas por clase (clase\_a, clase\_b, ...).

**Detección (YOLO):** imágenes con anotaciones en formato .txt (coordenadas normalizadas).

**Segmentación:** imágenes y máscaras binarias o multiclase en formato .png.

Las librerías principales son: `numpy`, `pandas`, `matplotlib`, `opencv-python`, y `Pillow`.

#### 3.2 PREPARACIÓN DEL ENTORNO

Incluye la opción de montar Google Drive (en caso de usar Google Colab).

Define rutas a los datasets y verifica qué clases están disponibles.

Se genera una estructura 20/80 para entrenamiento y validación, asegurando que los datos estén balanceados y limpios.

#### 3.3 CREACIÓN DEL DATASET

Se eliminan datasets previos si existen (limpieza).

Se crean nuevas carpetas para train y test.

Se implementa una función para copiar imágenes, dividir las según proporción definida y validar la consistencia.

#### 3.4 VISUALIZACIÓN INICIAL

Se muestra un ejemplo de imágenes por clase usando matplotlib.

Cada clase se representa con una imagen de muestra, lo que permite verificar:


- ✓ Calidad visual.
- ✓ Diferencias entre categorías.
- ✓ Posibles inconsistencias en el dataset.

#### 3.5 IMPORTACIÓN DE LIBRERÍAS Y ANÁLISIS

En secciones posteriores (según el índice detectado), se importan librerías adicionales para:

- ✓ **EDA cuantitativo:** conteo de imágenes por clase, tamaños de archivo, dimensiones.
- ✓ **EDA visual:** histogramas, distribución de proporciones train/test, ejemplos de bounding boxes (detección) y máscaras (segmentación).

### 3.6 TABLA COMPARATIVA DE MODOS

 Tabla descriptiva de modos de dataset en visión por computador

Modo	Estructura de carpetas	Qué contiene	Análisis EDA aplicado	Aplicación en proyecto (apilamiento de panel)
Clasificación	data/clase_a/*.jpg data/clase_b/*.jpg	Imágenes divididas en carpetas según clase (ej. OK vs Defecto)	<ul style="list-style-type: none"> <li>- Conteo de imágenes por clase (balance/desbalance).</li> <li>- Distribución de tamaños y resoluciones.</li> <li>- Histogramas de brillo y blur.</li> <li>- Detección de duplicados (pHash).</li> <li>- Outliers (imágenes borrosas o sobreexpuestas).</li> </ul>	Útil para un modelo CNN de clasificación que de paneles es correcto o incorrecto.
Detección (YOLO)	data/images/*.jpg data/labels/*.txt classes.txt (opcional)	Imágenes y anotaciones .txt con bounding boxes normalizados (class x y w h).	<ul style="list-style-type: none"> <li>- Distribución de objetos por imagen.</li> <li>- Conteo de clases en etiquetas.</li> <li>- Histogramas de tamaño relativo de bounding boxes.</li> <li>- Validación de etiquetas (coordenadas dentro de [0,1]).</li> <li>- Visualización con cajas dibujadas.</li> </ul>	Útil si quieres localizar defectos específicos (ej. huecos en la pila). El sistema no solo clasifica, también está el error.
Segmentación	data/images/*.jpg data/masks/*.png	Imágenes y sus máscaras (por píxel) indicando regiones de interés.	<ul style="list-style-type: none"> <li>- Cobertura de máscaras (proporción de píxeles etiquetados).</li> <li>- Distribución de cobertura (outliers: máscaras casi vacías o casi llenas).</li> <li>- Correspondencia 1:1 imagen ↔ máscara.</li> <li>- Validación de duplicados, blur y brillo.</li> </ul>	Permite entrenar un modelo de segmentación que identifique zona por zona si los paneles están o si hay anomalías en la superficie.

Tabla 2: Comparativa de modos / FUENTE: Generado ChatGPT



## 4. ANALISIS DE RESULTADOS

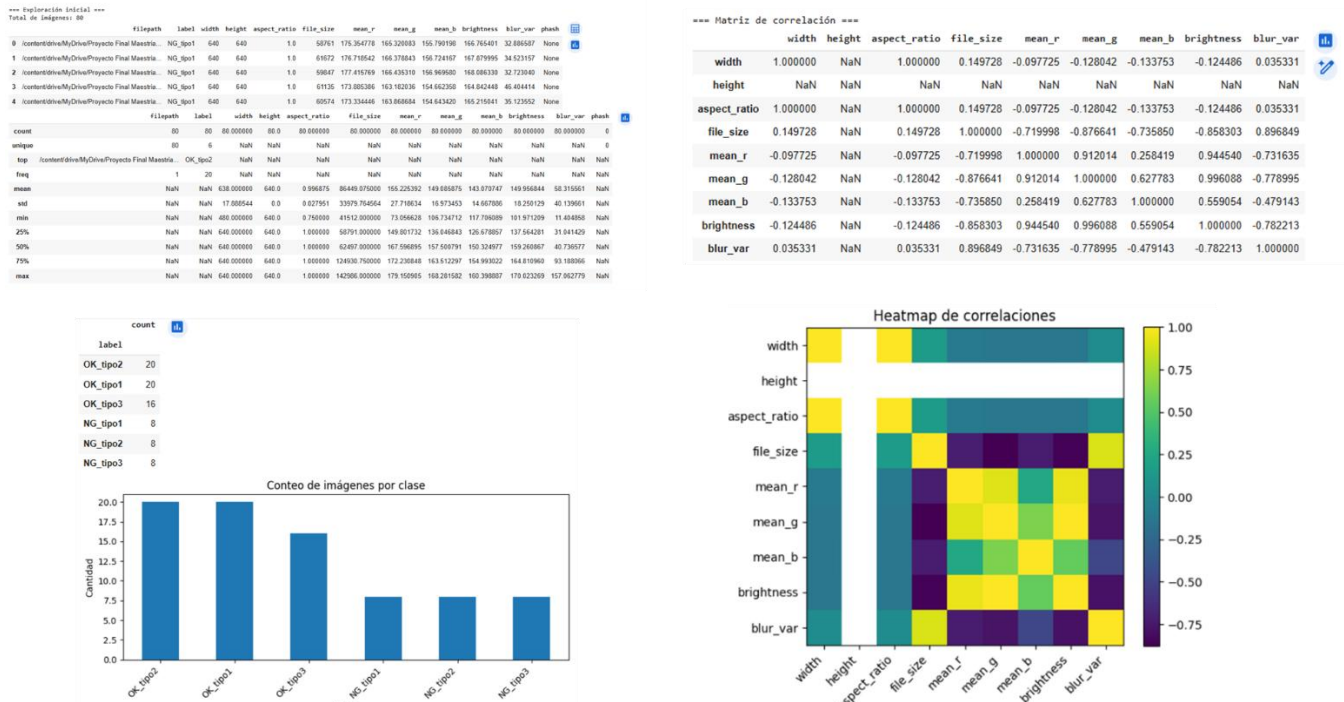


Figura 2: Resultados de NOTEBOOK / FUENTE: Notebook grupo 4  
Vision\_Computador\_Grupo\_4\_Rev\_5\_EDA.ipynb

### 4.1 EXPLORACIÓN INICIAL

El dataset analizado consta de 80 imágenes etiquetadas en seis clases (OK y NG con tres subtipos cada una). Cada imagen presenta resolución uniforme de 640x640 píxeles, con atributos asociados a dimensiones, tamaño de archivo, valores medios de canales RGB, brillo promedio y nivel de desenfoque (blur\_var). El análisis inicial muestra que las dimensiones no aportan variación, dado que el aspecto geométrico permanece constante en todas las instancias.

### 4.2 DISTRIBUCIÓN DE CLASES

El conteo por clases revela un desbalance significativo: mientras las clases OK\_tipo1 y OK\_tipo2 cuentan con 20 ejemplos cada una, las clases NG sólo contienen 8 ejemplos por subtipo. Esta situación puede sesgar los modelos predictivos hacia clasificar preferentemente la categoría mayoritaria. Se recomienda aplicar técnicas de balanceo como data augmentation en las clases minoritarias, o estrategias de oversampling.

### 4.3 CORRELACIONES Y PATRONES

La matriz de correlación evidencia relaciones fuertes entre las variables de color (mean\_r, mean\_g, mean\_b) y el brillo, lo que confirma redundancia de información cromática. Asimismo, el tamaño de archivo presenta alta correlación positiva con el brillo y negativa con el desenfoque. Este hallazgo sugiere que imágenes más nítidas tienden a ser menos luminosas y ocupar menor espacio de almacenamiento. Estas dependencias deben ser consideradas en la selección de características, evitando colinealidad y favoreciendo técnicas de reducción de dimensionalidad como PCA.

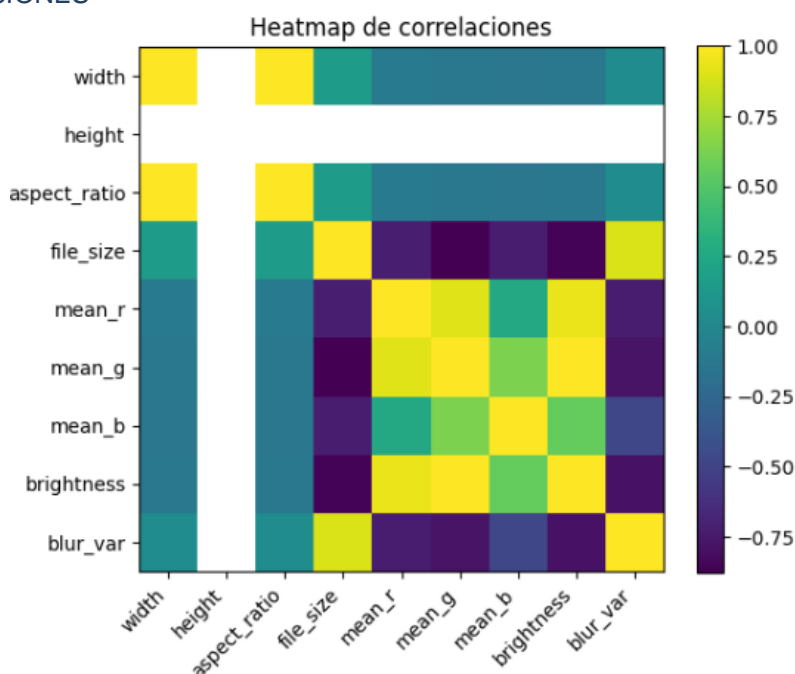
### 4.4 DISTRIBUCIONES ANÓMALAS E INCONSISTENCIAS

El análisis descriptivo muestra variabilidad en el brillo y en el desenfoque. En particular, la variable blur\_var evidencia casos de imágenes con alto nivel de desenfoque, lo cual puede impactar negativamente en la calidad del entrenamiento. La estandarización de la iluminación mediante normalización de histogramas y la eliminación o filtrado de imágenes excesivamente borrosas resultan pasos recomendables para fortalecer el dataset.

#### 4.5 INSIGHTS PRINCIPALES

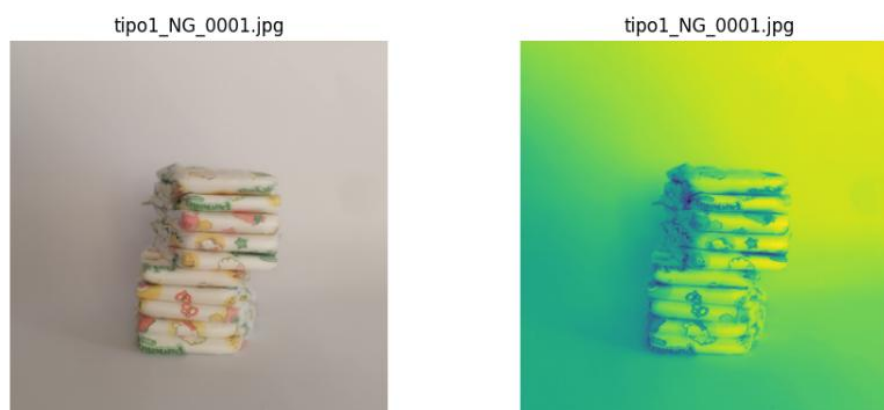
- ✓ El dataset presenta un desbalance de clases que debe ser corregido.
- ✓ Las variables cromáticas (RGB y brillo) son redundantes y pueden generar colinealidad.
- ✓ El desenfoque constituye un indicador relevante para diferenciar defectos (NG).
- ✓ La variabilidad de iluminación introduce ruido y debe ser mitigada con técnicas de normalización.

#### 4.6 CONCLUSIONES



*Figura 3: Resultados de NOTEBOOK / FUENTE: Notebook grupo 4  
Vision\_Computador\_Grupo\_4\_Rev\_5\_EDA.ipynb*

El análisis exploratorio permite concluir que el dataset, aunque útil como base experimental, requiere ajustes para su uso efectivo en entrenamiento de modelos de clasificación. La corrección del desbalance, la reducción de colinealidad y la normalización de condiciones de iluminación resultan prioritarias para garantizar un modelo robusto y generalizable. Este EDA constituye una etapa crítica que orienta el diseño de los siguientes pasos en el pipeline de visión por computador.



*Figura 4: Resultados de NOTEBOOK / FUENTE: Notebook grupo 4  
Vision\_Computador\_Grupo\_4\_Rev\_5\_EDA.ipynb*

El dataset de segmentación presenta una estructura clara y bien definida, lo que facilita su aplicación en modelos como U-Net o Mask R-CNN. Sin embargo, la variabilidad en desenfoque y las limitaciones en iluminación deben ser tratadas con técnicas de normalización y data augmentation. Este análisis confirma la utilidad del dataset para investigaciones académicas, destacando la importancia de un preprocesamiento cuidadoso antes del entrenamiento.

## 5. REFERENCIAS

- [1] N. Hütten, A. et al., “Deep Learning for Automated Visual Inspection in Industrial Applications,” *Machines*, vol. 12, no. 1, p. 11, Jan. 2024.
- [2] S. Arikan, K. Varanasi, and D. Stricker, “Surface Defect Classification in Real-Time Using Convolutional Neural Networks,” *arXiv preprint*, arXiv:1904.04671, 2019.
- [3] M. Jalayer, R. Jalayer, A. Kaboli, C. Orsenigo, and C. Vercellis, “Automatic Visual Inspection of Rare Defects: A Framework based on GP-WGAN and Enhanced Faster R-CNN,” *arXiv preprint*, arXiv:2105.00447, 2021.
- [4] T. Defard, A. Setkov, A. Loesch, and R. Audigier, “PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection and Localization,” *arXiv preprint*, arXiv:2011.08785, 2020.
- [5] T. Schlosser, F. Beuth, M. Friedrich, and D. Kowerko, “A Novel Visual Fault Detection and Classification System for Semiconductor Manufacturing Using Stacked Hybrid Convolutional Neural Networks,” *arXiv preprint*, arXiv:1911.11250, 2019.
- [6] J. Villalba-Diez et al., “Deep Learning for Industrial Computer Vision Quality,” *Sensors*, vol. 19, no. 22, pp. 4780, Nov. 2019.
- [7] A. Wan et al., “Deep learning-based intelligent visual inspection for defect detection in smoke sensor manufacturing,” *Engineering Applications of Artificial Intelligence*, vol. 142, p. 107004, Jan. 2025.
- [8] J. Huang, “Automated Logistics Packaging Inspection Based on Deep Learning and Computer Vision: A Two-Dimensional Flow Model Approach,” *Traitement du Signal*, vol. 42, no. 2, pp. 275–282, 2025.
- [9] “Advancing industrial inspection with an automated visual inspection tool powered by computer vision,” *Scientific Reports*, vol. 15, art. no. 88974, Feb. 2025.
- [10] C. D. Hollander, S. Das, and S. Suliman, “Automating Visual Inspection with Convolutional Neural Networks,” in *Proc. PHM Society Conf.*, 2019, pp. 868–875.
- [11] S. Yadav and R. Kennedy, “Vision Inspection Using Machine Learning/Artificial Intelligence,” *Pharmaceutical Engineering*, Nov. 2020.
- [12] R. A. León, et al., “Design of an artificial vision algorithm to detect cracks in construction,” in *Proc. LACCEI Int. Conf.*, Buenos Aires, 2023, pp. 1–9.
- [13] O. Barahona, “Design and Implementation of an Artificial Vision System for Defect Detection,” *KnE Engineering*, 2022.
- [14] M. C. Nava, “CNN computer for high-speed visual inspection,” in *Proc. SPIE 4301, VLSI Circuits and Systems*, 2001.
- [15] M. E. A. Macías et al., “Design and simulation of an artificial vision sorting machine and cocoa pulping machine,” *Ciencia Latina*, vol. 6, no. 3, pp. 1456–1468, 2022.
- [16] T. Defard et al., “Anomaly detection in industrial applications with deep learning,” *arXiv preprint*, arXiv:2011.08785, 2020.
- [17] M. Jalayer et al., “Faster R-CNN with synthetic data generation for rare defect inspection,” *arXiv preprint*, arXiv:2105.00447, 2021.
- [18] J. Huang, “Deep learning for logistics packaging inspection,” *Traitement du Signal*, vol. 42, no. 2, pp. 275–282, 2025.
- [19] J. Villalba-Diez et al., “Deep Learning for Industrial Quality Control,” *Sensors*, vol. 19, no. 22, pp. 4780, 2019.
- [20] A. Wan et al., “Visual inspection with DL in sensor manufacturing,” *Eng. Appl. Artif. Intell.*, vol. 142, p. 107004, 2025.
- [21] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 3rd ed. O'Reilly Media, 2022.

- [22] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.
- [23] J. C. Cuevas-Tello, "Apuntes de Redes Neuronales Artificiales," arXiv preprint, arXiv:1806.05298, 2018.
- [24] S. Soatto, Foundations of Computer Vision (Adaptive Computation and Machine Learning series). Cambridge, MA: MIT Press, 2024.
- [25] D. Forsyth and J. Ponce, Computer Vision: A Modern Approach, 2nd ed. Pearson, 2011.
- [26] Y. Goldberg, Neural Network Methods for Natural Language Processing. Morgan & Claypool, 2017.
- [27] F. Pasquale, Las nuevas leyes de la robótica. Madrid: Ediciones Paidós, 2024.