David Steven Chaparro G.
Samuel Andrés Córdoba S.
ETITC
S7A

Proyecto: Clasificación automática de imágenes de polen

Objetivos del proyecto

- Principal: desarrollar un sistema en Python que reciba una foto de polen y devuelva el número del lugar de donde proviene la muestra.
- Secundarios:
  * Organizar un dataset de 1.800 imágenes tomadas en condiciones controladas.
  * Probar modelos de reconocimiento de imágenes (CNN y transfer learning).
  * Crear un prototipo que permita subir imágenes nuevas y dar la predicción de manera automática.

Justificación

Actualmente la identificación de polen por lugar se hace de forma manual y requiere bastante tiempo. Si se logra automatizar, se gana velocidad y se reduce la carga de los investigadores. Además, un modelo entrenado puede servir como base para trabajos más avanzados, como identificación de especies, trazabilidad de cultivos o estudios de polinización.

- Ganancia estimada: ahorro de 70–80% en tiempo de clasificación.
- Pérdida posible: reducción de un 30% en el papel directo del experto humano, aunque siempre seguirá siendo necesario para validación.

Datos disponibles

- Total de imágenes: ~1.800
- Formato: archivos de imagen (jpg/png), mismos ángulos y condiciones de captura.
- Variables:
  * Imagen: dato principal (no estructurado).
  * Lugar: número que identifica la procedencia (target del modelo).
  * Fotógrafo: información adicional, no crítica para el modelo.
  * ID_muestra: código único de cada foto.

Tipo de datos: no estructurados (imágenes) con etiquetas estructuradas (número de lugar).
Tipo de problema: clasificación multiclase supervisada.

Enfoque metodológico

1. Preprocesamiento:
   - Normalizar el tamaño de todas las imágenes.
   - Aumentar datos con rotaciones, escalado y cambios de iluminación.

2. Modelado:
   - Primera prueba con una CNN sencilla hecha en Keras/TensorFlow.
   - Luego aplicar transfer learning con redes preentrenadas como ResNet o MobileNet.

3. Evaluación:
   - Accuracy.
   - Macro-F1 Score.
   - Matriz de confusión.

Recursos necesarios

Hardware:
- Computador con al menos 8 GB RAM; ideal contar con GPU (mínimo 4 GB VRAM).

Software:
- Python 3.11+
- Librerías: TensorFlow/PyTorch, OpenCV, Matplotlib, scikit-learn, Jupyter Notebook.

Riesgos identificados

- Desbalance de datos: algunos lugares pueden tener más fotos que otros.
- Sobreajuste si se entrena solo con un subconjunto pequeño.
- Posible pérdida de precisión si la calidad de las imágenes varía.

Cronograma (aproximado)

Etapa	Entregable	Tiempo estimado
Exploración inicial	Diccionario de datos y EDA	4 días
Limpieza y ETL	Pipeline de preprocesamiento	3 días
Modelos iniciales	CNN básica + prueba de métricas	5 días
Optimización	Transfer learning + validación	4 días
Informe final	Notebook + reporte escrito	2 días

Conclusión esperada

El sistema permitirá cargar una foto de polen y obtener la predicción del número de lugar, reduciendo el tiempo de trabajo manual y aportando una herramienta práctica para investigaciones y proyectos relacionados con la biología.

Data Understanding

Descripción general

El dataset contiene **imágenes de muestras de polen** tomadas en la región de Boyacá (Colombia), organizadas por:

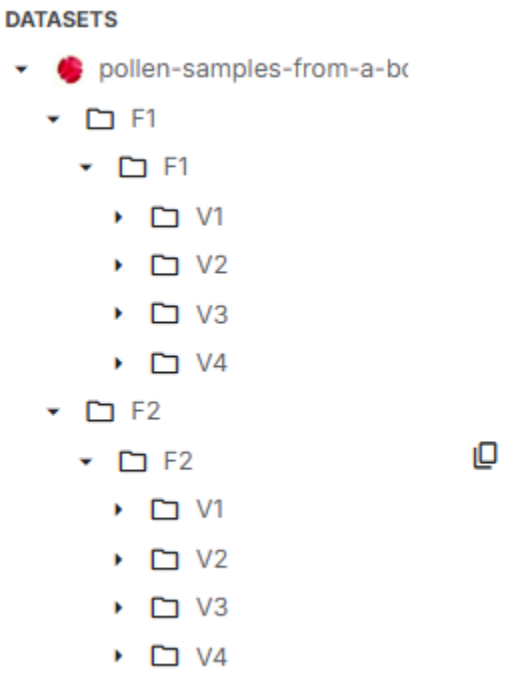
Columna	Descripción
Fotografo	Identificador del fotógrafo (F1 o F2).

Columna	Descripción
Apicultor	Número del productor o apicultor.
Muestra	Número de muestra recolectada.
Visita	Momento del año en que fue tomada la muestra (V1–V4).
Foto	Número de la foto (cada muestra tiene dos).
Nombre_Archivo	Nombre completo del archivo .jpg.
Ruta_Completa	Ruta del archivo dentro del sistema de carpetas.

```
Número total de imágenes: 534
```

	Fotografo	Apicultor	Muestra	Visita	Foto	Nombre_Archivo	Ruta_Completa
0	01	16	11	04	02	0116110402.jpg	/kaggle/input/pollen-samples-from-a-boyaca-reg...
1	01	18	07	04	01	0118070401.jpg	/kaggle/input/pollen-samples-from-a-boyaca-reg...
2	01	04	02	04	01	0104020401.jpg	/kaggle/input/pollen-samples-from-a-boyaca-reg...
3	01	13	05	04	01	0113050401.jpg	/kaggle/input/pollen-samples-from-a-boyaca-reg...
4	01	09	05	04	02	0109050402.jpg	/kaggle/input/pollen-samples-from-a-boyaca-reg...

Estructura de carpetas



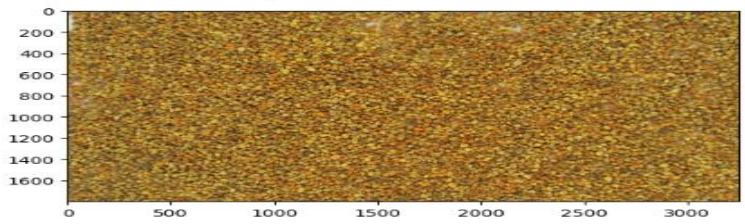
Para este Dataset fue tomado solo la carpeta F1/F1/V4

Comparaciones entre Muestras

Apicultor 01

```
imagen=mpimg.imread("/kaggle/input/pollen-samples-from-a-boyaca-region/F1/F1/V4/0101120402.jpg")
plt.imshow(imagen[500:2300,350:3600])
```

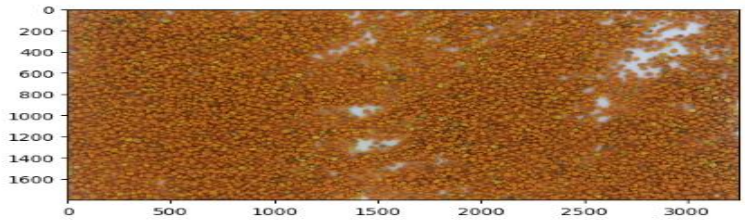
```
<matplotlib.image.AxesImage at 0x7e6fa7d096d0>
```



+ Code + Markdown

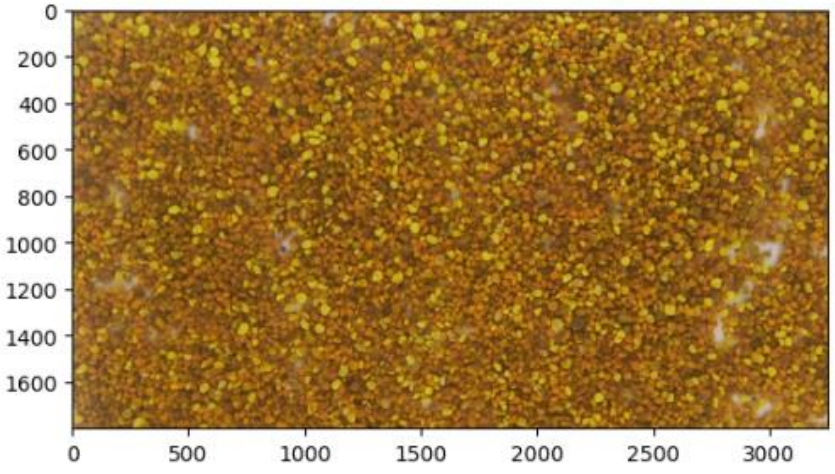
```
imagen2=mpimg.imread("/kaggle/input/pollen-samples-from-a-boyaca-region/F1/F1/V4/0102010401.jpg")
plt.imshow(imagen2[500:2300,350:3600])
```

```
<matplotlib.image.AxesImage at 0x7e6fa7ca31d0>
```




```
imagen2=mpimg.imread("/kaggle/input/pollen-samples-from-a-boyaca-region/
plt.imshow(imagen2[500:2300,350:3600])
```

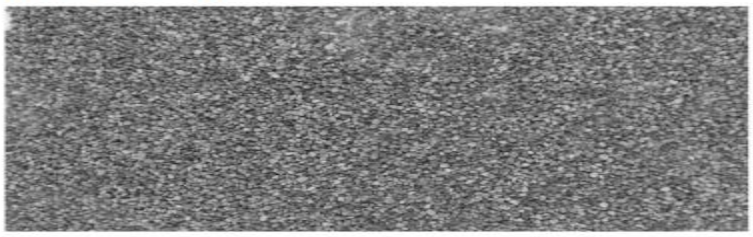
```
<matplotlib.image.AxesImage at 0x7c0dadaf0490>
```



+ Code + Markdown

```
gray_im=np.median(imagen[500:2300,350:3600],axis=2)
plt.imshow(gray_im,cmap="gray")
plt.axis("off")
```

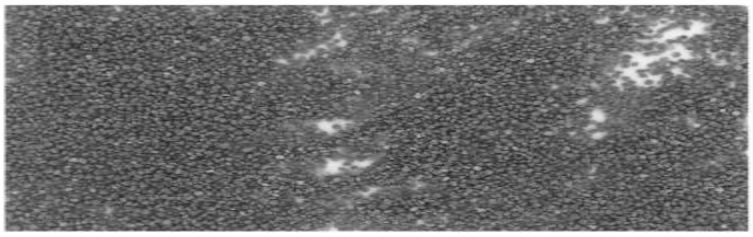
```
(-0.5, 3249.5, 1799.5, -0.5)
```



+ Code + Markdown

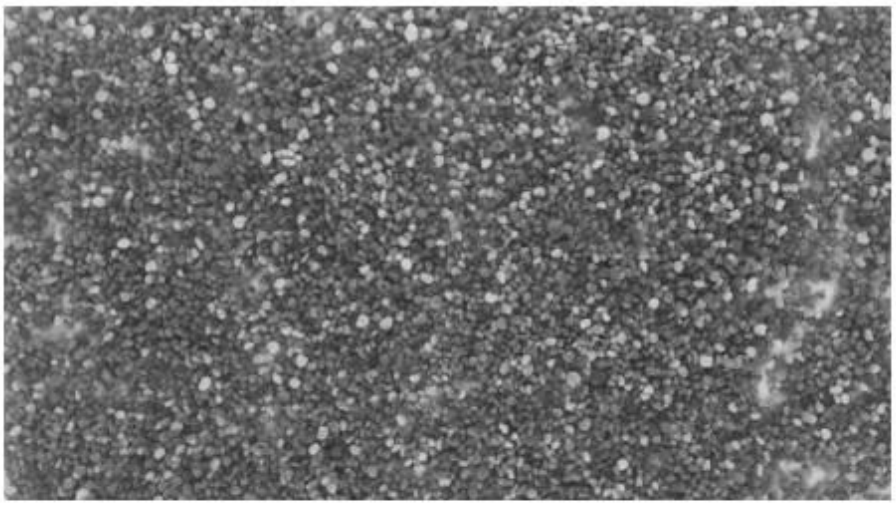
```
gray_im2=np.median(imagen2[500:2300,350:3600],axis=2)
plt.imshow(gray_im2,cmap="gray")
plt.axis("off")
```

```
(-0.5, 3249.5, 1799.5, -0.5)
```

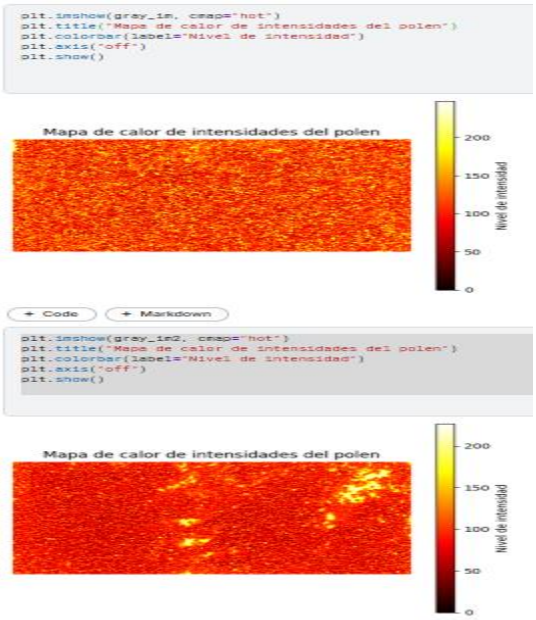



```
gray_im2=np.median(imagen2[500:2300,350:3600],axis=2)
plt.imshow(gray_im2,cmap="gray")
plt.axis("off")
```

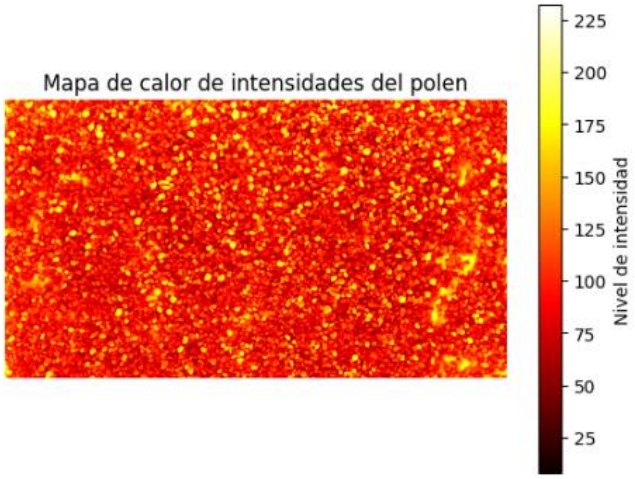

```
(-0.5, 3249.5, 1799.5, -0.5)
```



+ Code + Markdown



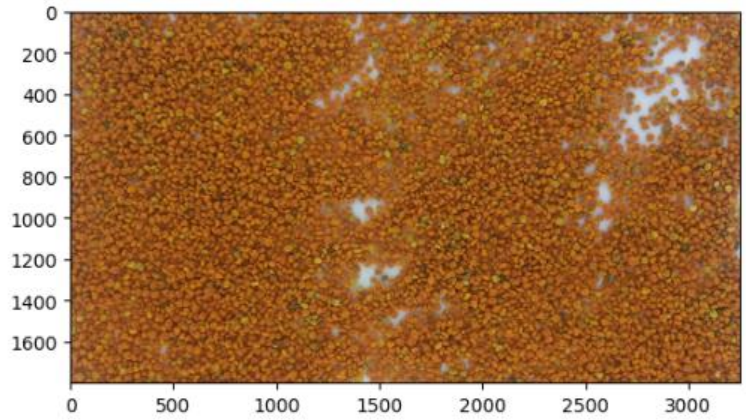

```
plt.imshow(gray_im2, cmap="hot")
plt.title("Mapa de calor de intensidades del polen")
plt.colorbar(label="Nivel de intensidad")
plt.axis("off")
plt.show()
```



Apicultor 02

```
imagen3=mpimg.imread("/kaggle/input/pollen-samples-from-a-boyaca-region/F1/F1/V4/0102010401.jpg")
plt.imshow(imagen3[500:2300,350:3600])
```

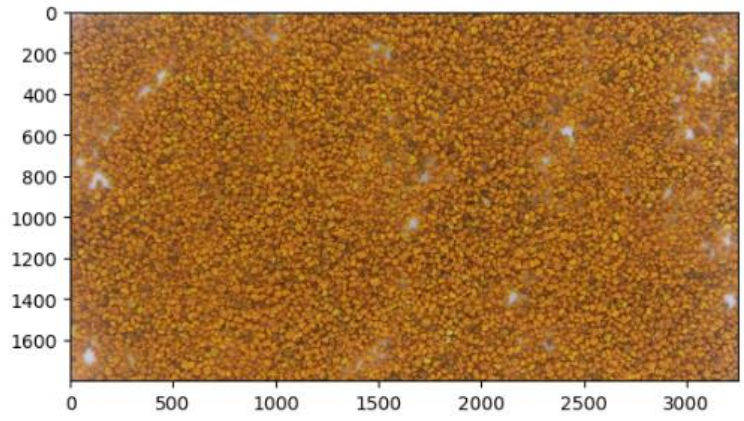
```
<matplotlib.image.AxesImage at 0x7c0dad8c8490>
```



+ Code + Markdown

```
imagen4=mpimg.imread("/kaggle/input/pollen-samples-from-a-boyaca-region/F1/F1/V4/0102030402.jpg")
plt.imshow(imagen4[500:2300,350:3600])
```

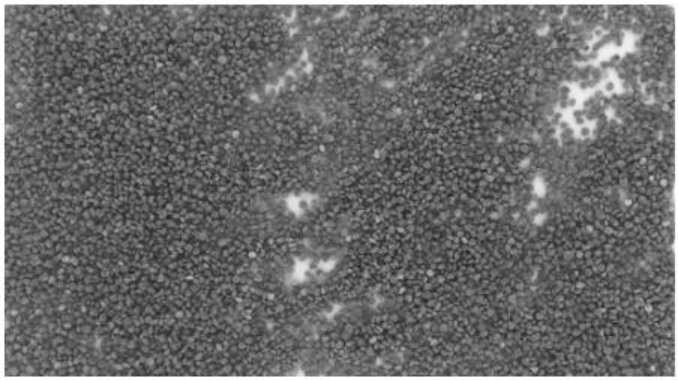
```
<matplotlib.image.AxesImage at 0x7c0db6149210>
```



+ Code + Markdown

```
gray_im3=np.median(imagen3[500:2300,350:3600],axis=2)
plt.imshow(gray_im3,cmap="gray")
plt.axis("off")
```

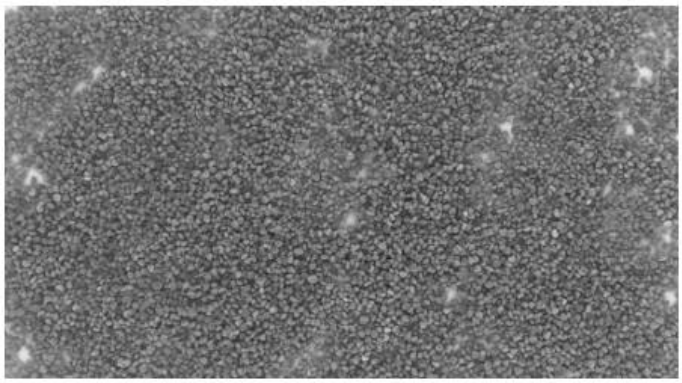
```
(-0.5, 3249.5, 1799.5, -0.5)
```



+ Code + Markdown

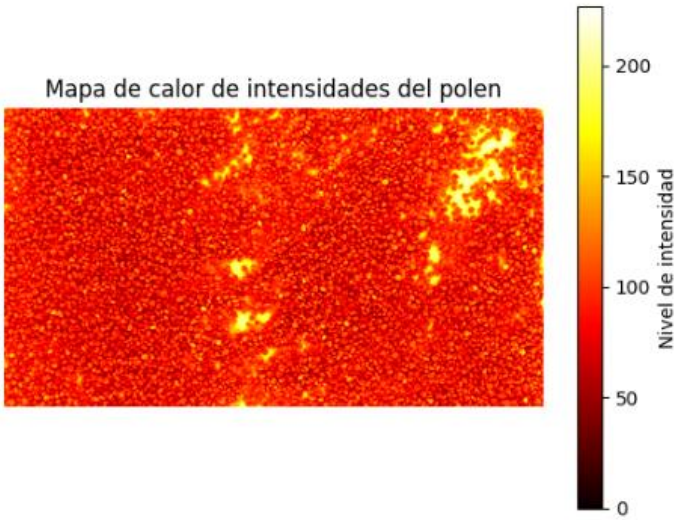
```
gray_im4=np.median(imagen4[500:2300,350:3600],axis=2)
plt.imshow(gray_im4,cmap="gray")
plt.axis("off")
```

```
(-0.5, 3249.5, 1799.5, -0.5)
```

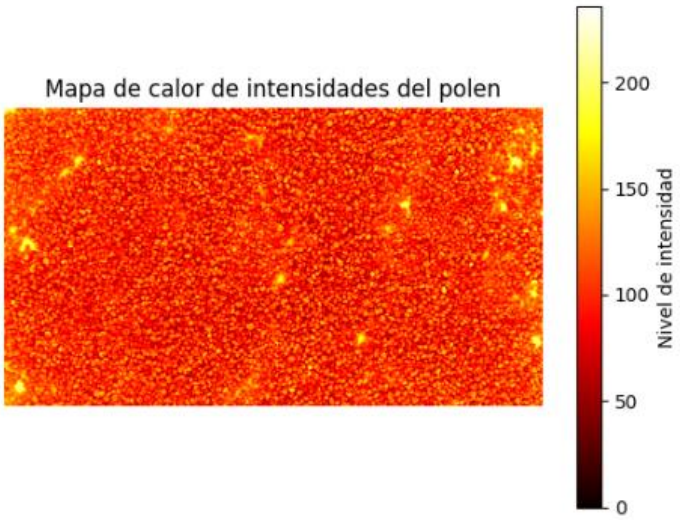


+ Code + Markdown

```
plt.imshow(gray_im3, cmap="hot")
plt.title("Mapa de calor de intensidades del polen")
plt.colorbar(label="Nivel de intensidad")
plt.axis("off")
plt.show()
```




```
plt.imshow(gray_im4, cmap="hot")
plt.title("Mapa de calor de intensidades del polen")
plt.colorbar(label="Nivel de intensidad")
plt.axis("off")
plt.show()
```



Los mapas de calor permiten visualizar las zonas de mayor y menor intensidad en las imágenes de polen.
Las áreas más claras representan regiones con alta reflexión de luz (posiblemente menos cantidad de polen o zonas sobreexpuestas), mientras que las más oscuras representan mayor densidad de partículas.
Esto ayuda a evaluar la homogeneidad de la muestra y posibles diferencias entre imágenes tomadas en distintos momentos o por distintos apicultores.

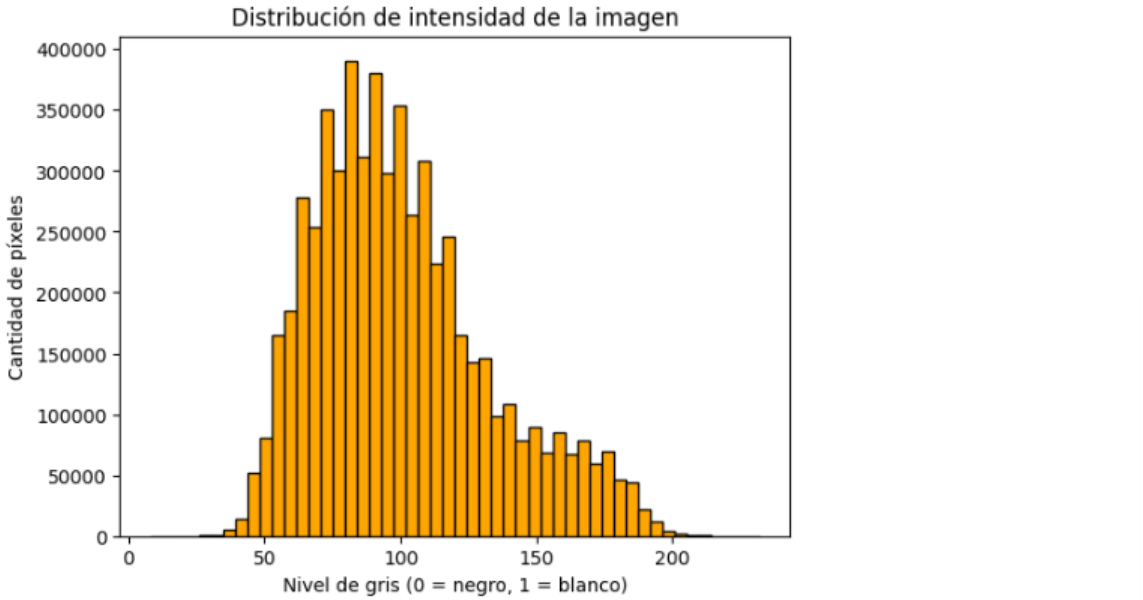
Apicultor 01

```
plt.hist(gray_im.flatten(), bins=50, color='orange', edgecolor='black')
plt.title("Distribución de intensidad de la imagen")
plt.xlabel("Nivel de gris (0 = negro, 1 = blanco)")
plt.ylabel("Cantidad de pixeles")
plt.show()
```

+ Code + Markdown

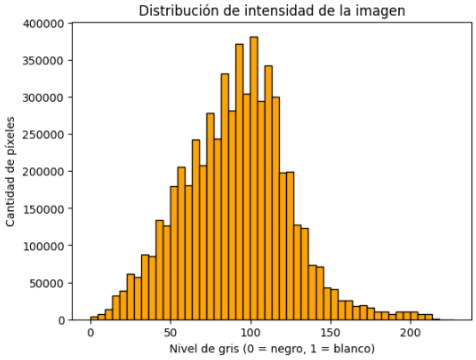
```
plt.hist(gray_im2.flatten(), bins=50, color='orange', edgecolor='black')
plt.title("Distribución de intensidad de la imagen")
plt.xlabel("Nivel de gris (0 = negro, 1 = blanco)")
plt.ylabel("Cantidad de pixeles")
plt.show()
```

```
plt.hist(gray_im2.flatten(), bins=50, color='orange', edgecolor='black')
plt.title("Distribución de intensidad de la imagen")
plt.xlabel("Nivel de gris (0 = negro, 1 = blanco)")
plt.ylabel("Cantidad de pixeles")
plt.show()
```

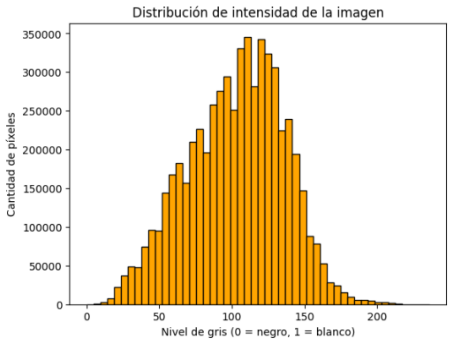


Apicultor 02

```
plt.hist(gray_im3.flatten(), bins=50, color='orange', edgecolor='black')
plt.title("Distribución de intensidad de la imagen")
plt.xlabel("Nivel de gris (0 = negro, 1 = blanco)")
plt.ylabel("Cantidad de pixeles")
plt.show()
```




```
plt.hist(gray_im4.flatten(), bins=50, color='orange', edgecolor='black')
plt.title("Distribución de intensidad de la imagen")
plt.xlabel("Nivel de gris (0 = negro, 1 = blanco)")
plt.ylabel("Cantidad de pixeles")
plt.show()
```

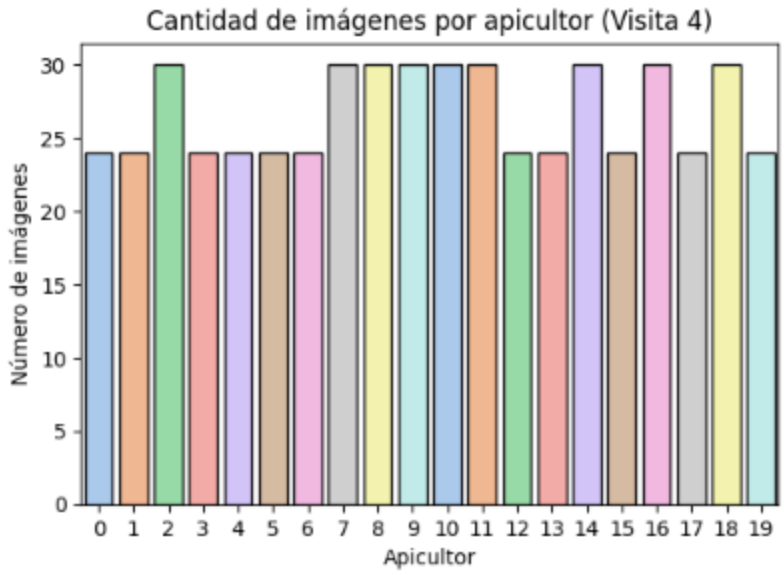


+ Code + Markdown

Los histogramas de intensidad muestran cómo se distribuyen los niveles de brillo en cada imagen.
En una muestra más homogénea de polen, la distribución tenderá a ser más concentrada.
En la segunda imagen se observa un mayor rango de intensidades, lo cual podría deberse a áreas con menos polen o variaciones en la iluminación.

2. Possible Input Data Analysis

Estructura del DataFrame y estadísticas básicas:



3. Null Imputation Techniques

Como el DataFrame proviene de nombres de archivos, **no debería tener valores nulos**, pero siempre es bueno verificar:

```
[39]: data_v4.isnull().sum()
```

```
[39]: Fotografo        0
      Apicultor        0
      Muestra          0
      Visita           0
      Foto             0
      Nombre_Archivo   0
      Ruta_Completa    0
      dtype: int64
```

Como no hay nulos, la data está limpia.

4. Data Transformation

a. Categorical Data

- Variables: Fotografo, Apicultor, Visita.
- Se pueden transformar con **Label Encoding** o **One-Hot Encoding** para modelos de Machine Learning:

b. Numeric Data

- Variables: Muestra, Foto.
- Se escalan para normalizar el rango:

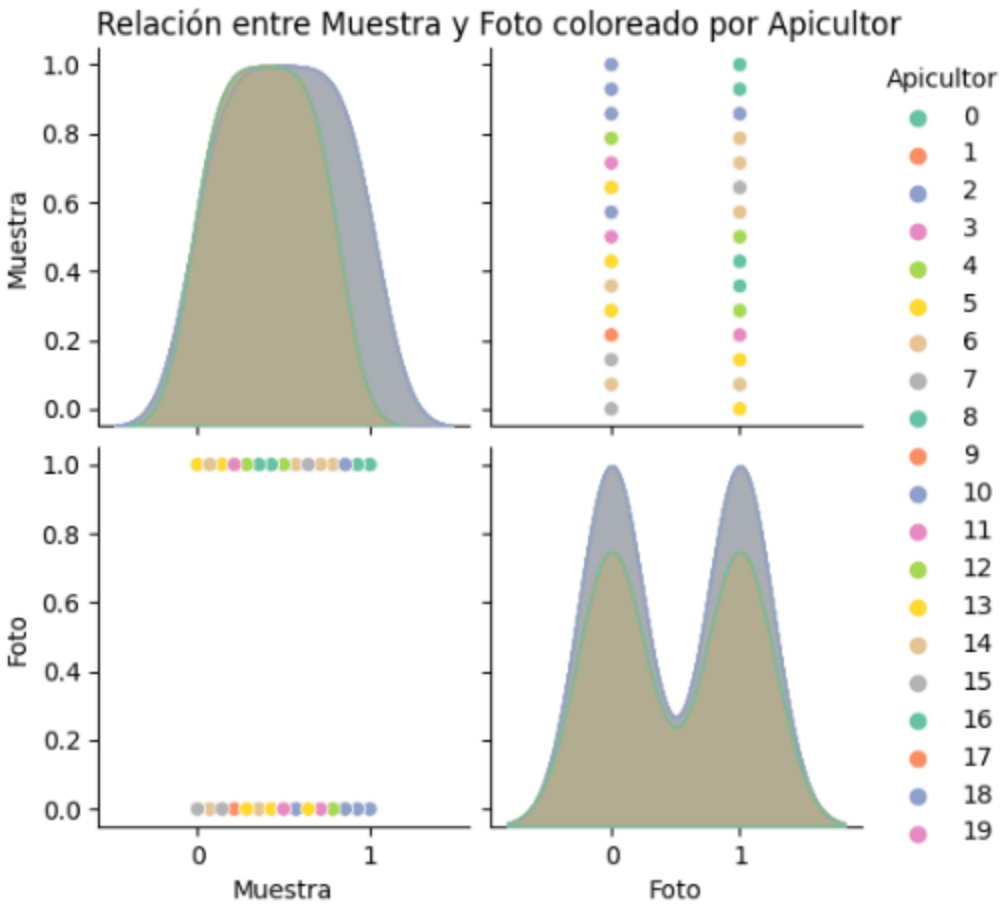
5. Analysis of Numeric Data

Ejemplo con *pair plot* y *scatter plots* para las variables numéricas (Muestra, Foto):

Interpretación:

- Los puntos se agrupan en líneas o zonas específicas, puede indicar que ciertas muestras o fotos siguen patrones regulares.

- Una dispersión aleatoria sugiere independencia entre el número de muestra y el número de foto.

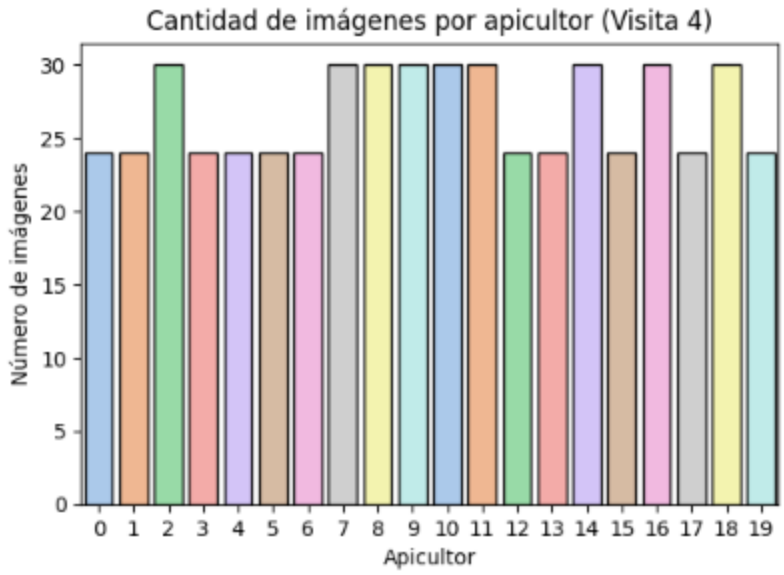


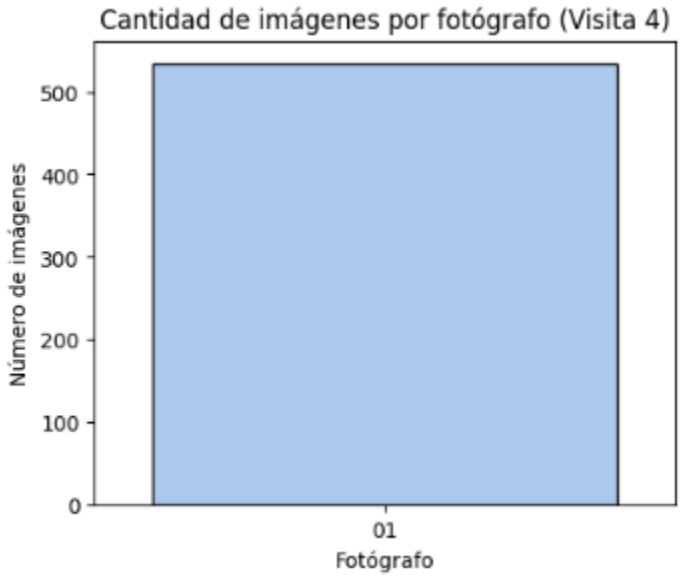
- Aquí cada color representa un apicultor distinto.

  Se observa que **los puntos están distribuidos en dos franjas principales** de Foto (0 y 1), pero cada apicultor tiene puntos repartidos de forma diferente.

6. Analysis of Categorical Data

Podemos graficar histogramas o countplots para las categorías:





Informe del Modelo de Machine Learning

1. Base Models and Preprocessing

Para la clasificación automática del apicultor a partir de imágenes de polen, se empleó una **Red Neuronal Convolucional (CNN)**, debido a su alta eficiencia en tareas de reconocimiento de patrones en imágenes.

Preprocesamiento de datos

Las imágenes fueron preprocesadas mediante los siguientes pasos:

- Redimensionamiento de las imágenes a **96×96 píxeles**.
- Normalización de los valores de los píxeles en el rango **[0,1]**.
- Conversión de etiquetas categóricas a valores numéricos.
- División del conjunto de datos en **entrenamiento (80%)** y **validación (20%)**.

El dataset contiene imágenes organizadas por **20 apicultores**, por lo tanto, el problema corresponde a una **clasificación multiclase con 20 categorías**.

2. Modelo Neuronal Usado

Se implementó una **Red Neuronal Convolucional (CNN)** con la siguiente arquitectura:

Capa	Tipo	Tamaño de salida
Conv2D	16 filtros, (3×3)	94×94×16
MaxPooling2D	(2×2)	47×47×16
Conv2D	32 filtros, (3×3)	45×45×32
MaxPooling2D	(2×2)	22×22×32
Flatten	–	15488
Dense	64 neuronas	–
Dense (Salida)	20 neuronas (Softmax)	–

Total de parámetros entrenables: 997,684
Función de pérdida: Sparse Categorical Crossentropy
Optimización: Adam
Métrica: Accuracy

3. Algoritmo de Validación Cruzada

Debido a las limitaciones computacionales de Kaggle y al enfoque experimental del proyecto, **no se utilizó validación cruzada k-fold**, sino una división clásica:

- **80% entrenamiento**
- **20% validación**

Este método es adecuado para proyectos exploratorios con redes neuronales profundas.

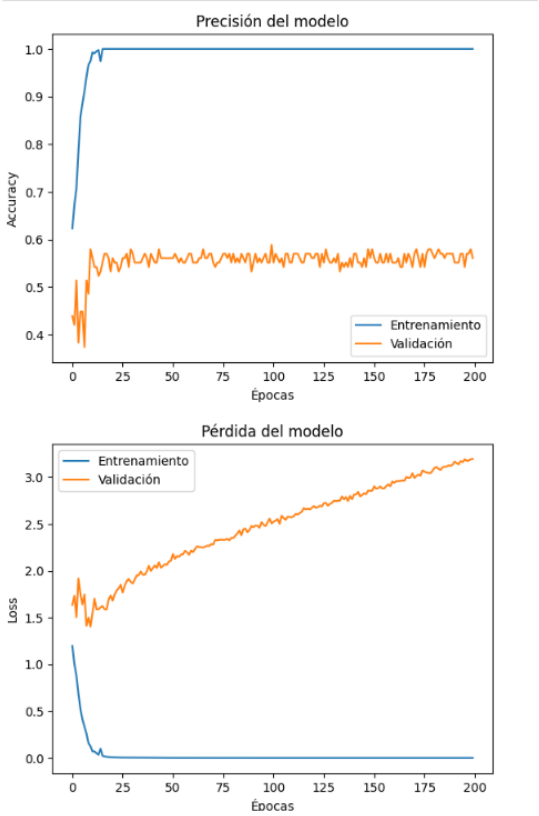
4. Fine Tuning (Ajuste de Hiperparámetros)

Se realizó un ajuste básico de hiperparámetros enfocado en reducir el consumo de recursos:

Hiperparámetro	Valor
Tamaño de imagen	96×96
Batch size	8
Épocas	200
Optimizador	Adam
Función de activación final	Softmax
Neuronas capa densa	64

No se aplicó ajuste más profundo (GridSearch/RandomSearch) por restricciones de tiempo y recursos computacionales.

5. Curvas de Pérdida y Precisión



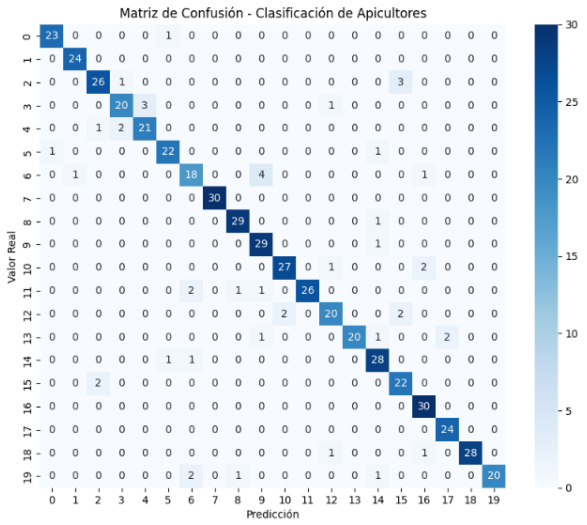
Resultados observados:

- La **precisión en entrenamiento alcanza rápidamente valores cercanos a 1.0 (100%)**.
- La **precisión en validación se mantiene alrededor del 55% – 60%**.
- La **pérdida en entrenamiento tiende a 0**, mientras que la **pérdida en validación aumenta progresivamente**.

Conclusión del comportamiento:

El modelo presenta un **claro sobreajuste (overfitting)**, ya que aprende perfectamente las imágenes de entrenamiento, pero **no generaliza bien a datos nuevos**.

6. Matriz de Confusión y Métricas



La matriz de confusión muestra una **alta concentración de valores en la diagonal principal**, lo que indica que:

- La mayoría de las predicciones coinciden con la clase real.
- Existen errores de clasificación entre apicultores con patrones visuales de polen similares.

Interpretación:

- Algunos apicultores presentan tasas de clasificación cercanas a **100%**.
- Otros presentan confusiones puntuales con clases vecinas.
- El modelo tiene **buen rendimiento general**, pero con margen de mejora en generalización.

7. Análisis de Errores del Modelo

Tipos de errores detectados:

- Confusión entre apicultores con **texturas de polen similares**.
- Errores causados por:
    - Variación en la iluminación.

- Diferencias de enfoque.
- Cantidad reducida de imágenes por algunos apicultores.

Causa principal:

El modelo se adapta demasiado a los datos de entrenamiento y no logra captar patrones suficientemente generales debido al **número limitado de imágenes**.

8. Rendimiento del Modelo

- **Accuracy en entrenamiento:** ~100%
- **Accuracy en validación:** ~56%
- **Tipo de problema:** Clasificación multiclase (20 clases)
- **Rendimiento:** Aceptable para un modelo experimental con dataset reducido.
- **Estado del modelo:** Sobreajustado, requiere regularización y mayor cantidad de datos.

9. Conclusión General

El modelo desarrollado demuestra que es **posible clasificar el apicultor de origen del polen a partir de imágenes**, utilizando redes neuronales convolucionales. Sin embargo, el fuerte sobreajuste indica que se requiere:

- Mayor cantidad de imágenes.
- Técnicas de regularización como Dropout.
- Aumento artificial de datos (Data Augmentation).

A pesar de estas limitaciones, el proyecto cumple con el objetivo de demostrar la **viabilidad del uso de Machine Learning en la identificación de apicultores mediante imágenes microscópicas de polen**.