

# **FORMACIÓN PROFESIONAL**

## **CURSO DE PRÁCTICA INTENSIVA**

### **CUADERNO DE INFORMES**



DIRECCIÓN ZONAL

LAMBAYEQUE

## FORMACIÓN PROFESIONAL

CFP/UCP/ESCUELA: SENATI

ESTUDIANTE: Jhon Erick Rodríguez Monja

ID: 1535824 BLOQUE: Dual, Tec.Ind, Adm, T Ing 2024-2

CARRERA: Ingeniería de Software con Inteligencia Artificial

INSTRUCTOR: Eduardo Alfonso Torres Luna

SEMESTRE: V DEL: 09/09/24 AL: 12/09/24.



## INSTRUCCIONES PARA EL USO DEL CUADERNO DE INFORMES

### 1. PRESENTACIÓN.

El Cuaderno de Informes es un documento de auto control, en el cual el estudiante, registra diariamente, durante la semana, las tareas, operaciones que ejecuta en su aprendizaje, es un medio para desarrollar la Competencia de Redactar Informes.

### 2. INSTRUCCIONES PARA EL USO DEL CUADERNO DE INFORMES.

2.1 En la hoja de informe semanal, el estudiante registrará diariamente los trabajos que ejecuta, indicando el tiempo correspondiente. El día de asistencia registrará los contenidos que desarrolla. Al término de la semana totalizará las horas.

De las tareas ejecutadas durante la semana, el ESTUDIANTE seleccionará la tarea más significativa (1) y él hará una descripción del proceso de ejecución con esquemas, diagramas y dibujos correspondientes que aclaren dicho proceso.

2.2 Semanalmente, el Instructor revisará y calificará el Cuaderno de Informes haciendo las observaciones y recomendaciones que considere convenientes, en los aspectos relacionados a la elaboración de un Informe Técnico (letra normalizada, dibujo técnico, descripción de la tarea y su procedimiento, normas técnicas, seguridad, etc.

2.3 Escala de calificación vigesimal:

CUANTITATIVA	CUALITATIVA	CONDICIÓN
16,8 – 20,0	Excelente	Aprobado
13,7 – 16,7	Bueno	
10,5 – 13,6	Aceptable	
00 – 10,4	Deficiente	Desaprobado

## INFORME SEMANAL

.....IV....SEMESTRE      SEMANA N° 5

DÍA	TAREAS EFECTUADAS	HORAS
LUNES	<p><b><u>IMPORTACION DE LIBRERIAS KERAS</u></b></p> <ul style="list-style-type: none"><li>• Explicación sobre los <b>FUNDAMENTOS Y ALGORITMIA PARA INTELIGENCIA ARTIFICIAL</b></li><li>• Machine Learning</li><li>• Deep Learning</li><li>• Clasificación y regresión</li><li>• Regresión Lineal</li><li>• Regresión Logística</li></ul>	5
JUEVES	<p><b><u>REALIZAMOS UNA PRACTICA</u></b></p> <ul style="list-style-type: none"><li>• Clasificador -Binario.</li></ul>	7
TOTAL		12

## INFORME DE TAREA MÁS SIGNIFICATIVA

**Tarea Significativa:** Clasificación Binaria de Imágenes con Deep Learning

Importación de Librerías:

**Descripción del Proceso:**

- Importación de Librerías:
- Se importan las principales librerías necesarias para el desarrollo del modelo:
- Numpy: para el manejo de matrices y procesamiento numérico.
- TensorFlow/Keras: para construir, entrenar y cargar el modelo de deep learning.
- Matplotlib: para mostrar las imágenes y resultados.

**Clasificación Binaria de Imágenes:**

- El modelo clasifica imágenes en dos categorías: Clase A y Clase B.

**Carga del Modelo:**

- Se utiliza la función `load_model()` de Keras para cargar un modelo previamente entrenado almacenado en el archivo 'modelo.h5'.

**Función de Procesamiento de Imágenes:**

- Se define una función `preprocesar_imagen()` para cargar una imagen, redimensionarla al tamaño objetivo (ejemplo: 64x64 píxeles o el tamaño que el modelo espera), y normalizar los valores de los píxeles dividiendo por 255.

**Configuración de los Generadores de Imágenes:**

- Se pueden usar generadores como `ImageDataGenerator` en Keras para cargar imágenes desde directorios, ajustándolas automáticamente a un tamaño de 164x164 píxeles (o el tamaño que desees para el entrenamiento y validación del modelo).

**Carga y Preprocesamiento de la Imagen de Prueba:**

- Se carga la imagen que se desea clasificar y se procesa para ser compatible con el modelo (redimensionada y normalizada).

**Predicción:**

- Con el modelo ya cargado, se realiza una predicción usando el método `model.predict()` para la imagen procesada. La salida será un valor entre 0 y 1, representando la probabilidad de pertenecer a una clase u otra.

**Determinación del Resultado:**

- Si el valor de la predicción es mayor a 0.5, se clasifica la imagen como Clase B; de lo contrario, como Clase A.

### **Visualización de la Imagen y Resultados:**

- Se usa Matplotlib para mostrar la imagen junto con el nombre del archivo y la clase detectada superpuestas.

### **Función General para Entrenar y Usar el Modelo:**

- Todo este proceso permite entrenar un modelo de clasificación binaria de imágenes y luego utilizarlo para predecir la clase de nuevas imágenes, haciendo que el modelo sea reutilizable en distintos contextos de clasificación de imágenes.

<b>HACER ESQUEMA, DIBUJO O DIAGRAMA</b>
---

### **LUNES:**

#### **Fundamentos y Algoritmia para Inteligencia Artificial**

- La Inteligencia Artificial (IA) es un campo de la informática que se dedica a crear sistemas que pueden realizar tareas que normalmente requieren inteligencia humana, como reconocimiento de voz, toma de decisiones y traducción de lenguajes. Los fundamentos de la IA se basan en la capacidad de las máquinas para aprender de los datos, resolver problemas y mejorar con el tiempo. La algoritmia en IA implica diseñar y desarrollar algoritmos que permitan a las máquinas aprender de los datos, tomar decisiones y realizar predicciones.

### **Machine Learning**

- Machine Learning (ML) o aprendizaje automático es una rama de la IA que se centra en desarrollar algoritmos que permiten a las máquinas aprender patrones a partir de los datos y hacer predicciones sin ser programadas explícitamente para cada tarea. ML utiliza modelos matemáticos y estadísticos que se entrenan con datos históricos para realizar predicciones futuras.

## **Los tres tipos principales de ML son:**

- **Aprendizaje supervisado:** El algoritmo aprende a partir de datos etiquetados, es decir, se le da ejemplos de entradas con sus correspondientes salidas deseadas (como clasificación o regresión).
- **Aprendizaje no supervisado:** El algoritmo trabaja con datos no etiquetados, buscando patrones o agrupaciones inherentes (como clustering o reducción de dimensionalidad).
- **Aprendizaje por refuerzo:** El modelo aprende mediante la interacción con su entorno, tomando decisiones que maximizan una recompensa acumulada.

## **Deep Learning**

- **Deep Learning (DL)** es un subcampo del Machine Learning que utiliza redes neuronales artificiales profundas para modelar relaciones complejas entre los datos. En DL, las redes neuronales tienen múltiples capas (por eso se llama "profundo") que permiten representar estructuras y características a diferentes niveles de abstracción.
- Las redes neuronales profundas están compuestas por capas de neuronas artificiales (o nodos), cada una conectada a la siguiente capa mediante pesos que se ajustan durante el entrenamiento. DL es eficaz para tareas como reconocimiento de imágenes, procesamiento de lenguaje natural y sistemas de recomendación, ya que puede capturar patrones no lineales y altamente complejos en los datos.

## **Clasificación y Regresión**

- Tanto la clasificación como la regresión son problemas supervisados en Machine Learning.
- **Clasificación:** El objetivo es asignar una categoría o clase a una entrada basada en sus características. Ejemplos de clasificación incluyen determinar si un correo electrónico es spam o no, o predecir el género de una persona en función de sus hábitos de compra.

- Regresión: El objetivo es predecir un valor continuo. Por ejemplo, predecir el precio de una casa basada en sus características o el crecimiento de las ventas en el próximo trimestre.

## **Regresión Lineal**

- La regresión lineal es un algoritmo básico de Machine Learning utilizado para modelar la relación entre una variable dependiente (Y) y una o más variables independientes (X). El modelo asume que la relación entre las variables es lineal y busca ajustar una línea recta a los datos que minimice el error.

**La ecuación de la regresión lineal simple es:**

- $Y = \beta_0 + \beta_1 X + \epsilon$

## **Regresión Logística**

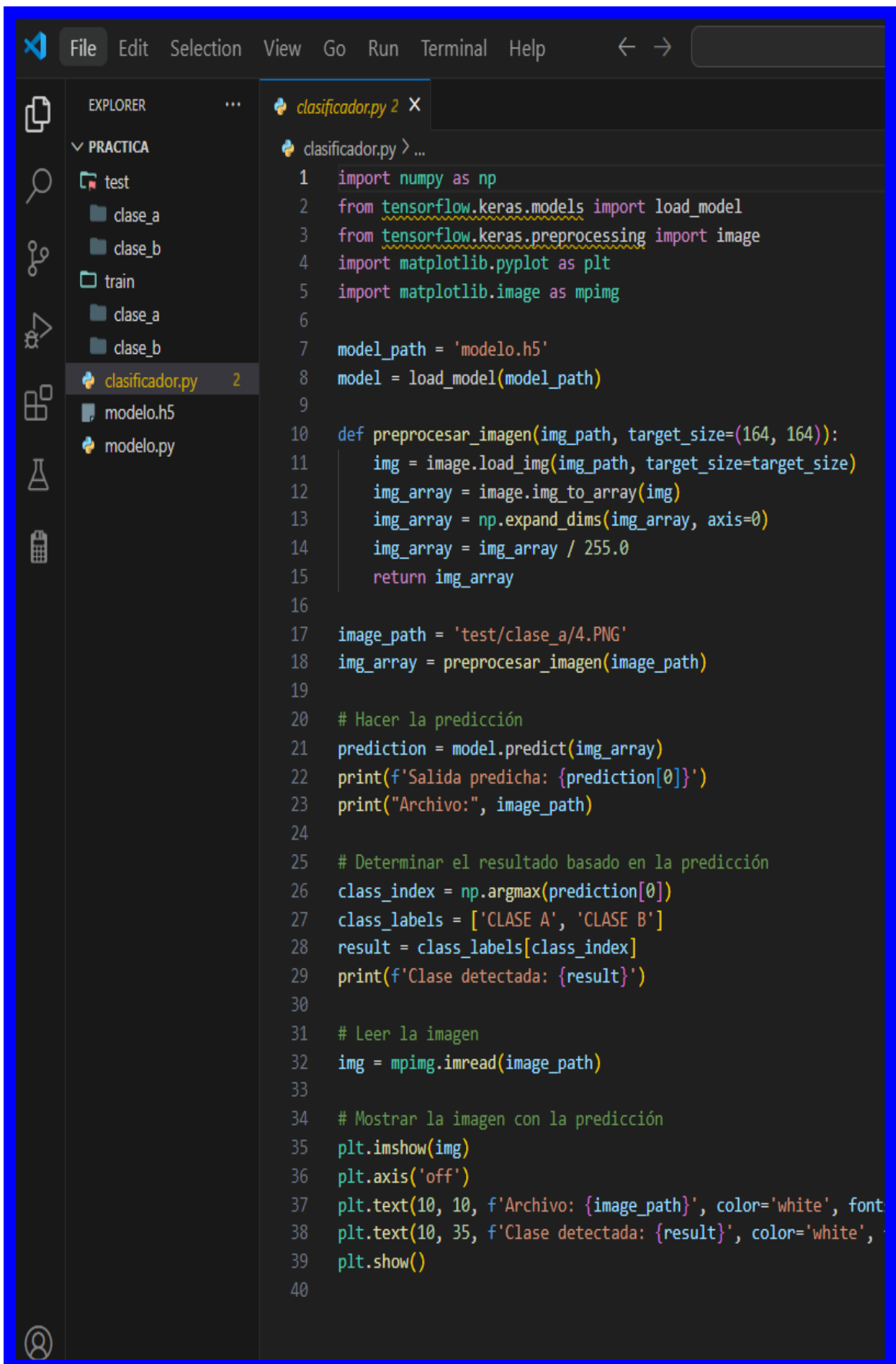
- La regresión logística es un algoritmo de clasificación que se utiliza para predecir la probabilidad de que una instancia pertenezca a una clase binaria (por ejemplo, 0 o 1). A diferencia de la regresión lineal, que predice valores continuos, la regresión logística estima una función sigmoide que convierte los valores predichos en probabilidades entre 0 y 1.

**La función sigmoide es:**

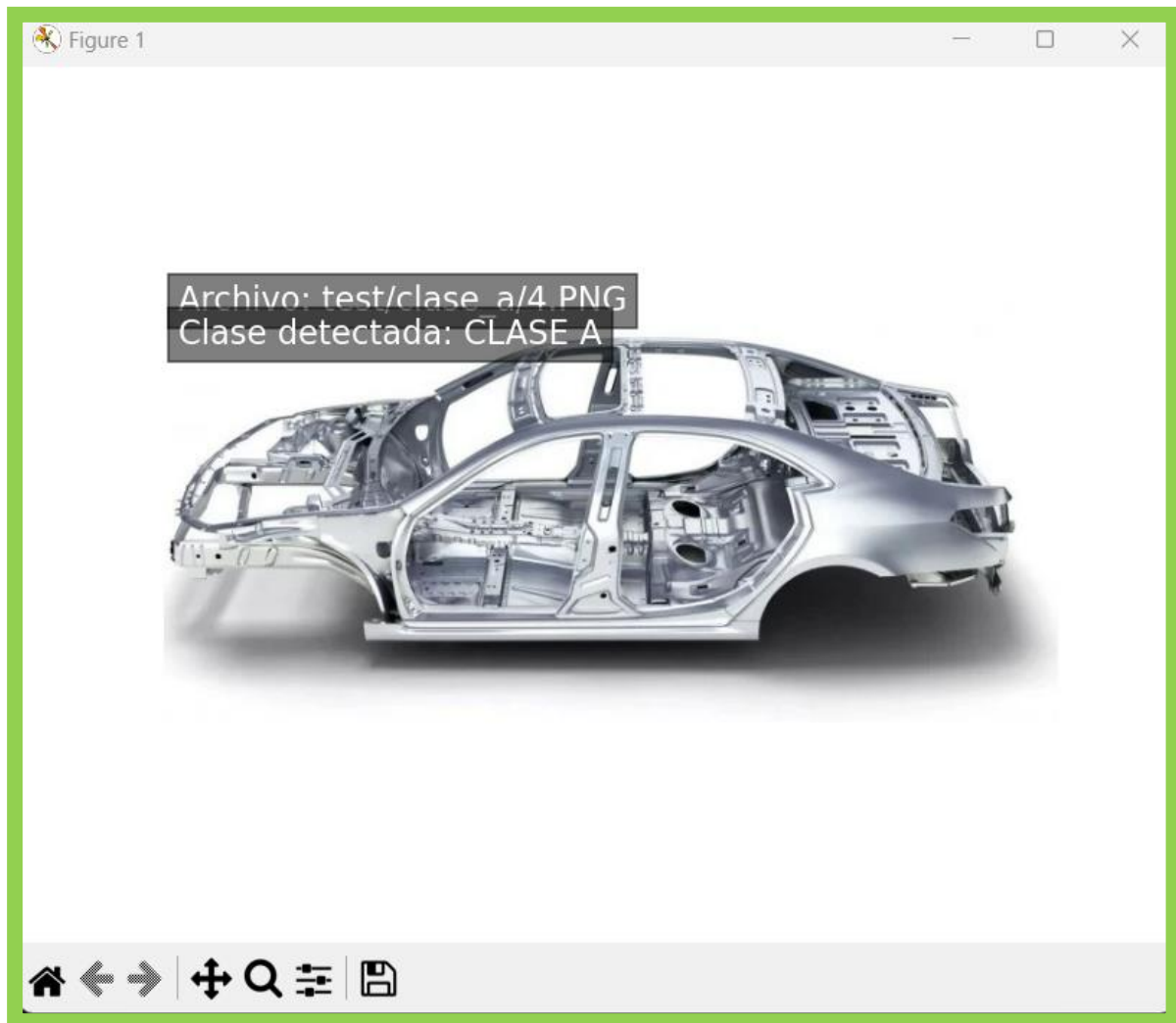
$$P(Y=1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

- Este modelo se utiliza para problemas donde el resultado es binario, como determinar si un cliente realizará una compra o no, o si un paciente tiene una enfermedad en función de ciertos síntomas.





## JUEVES



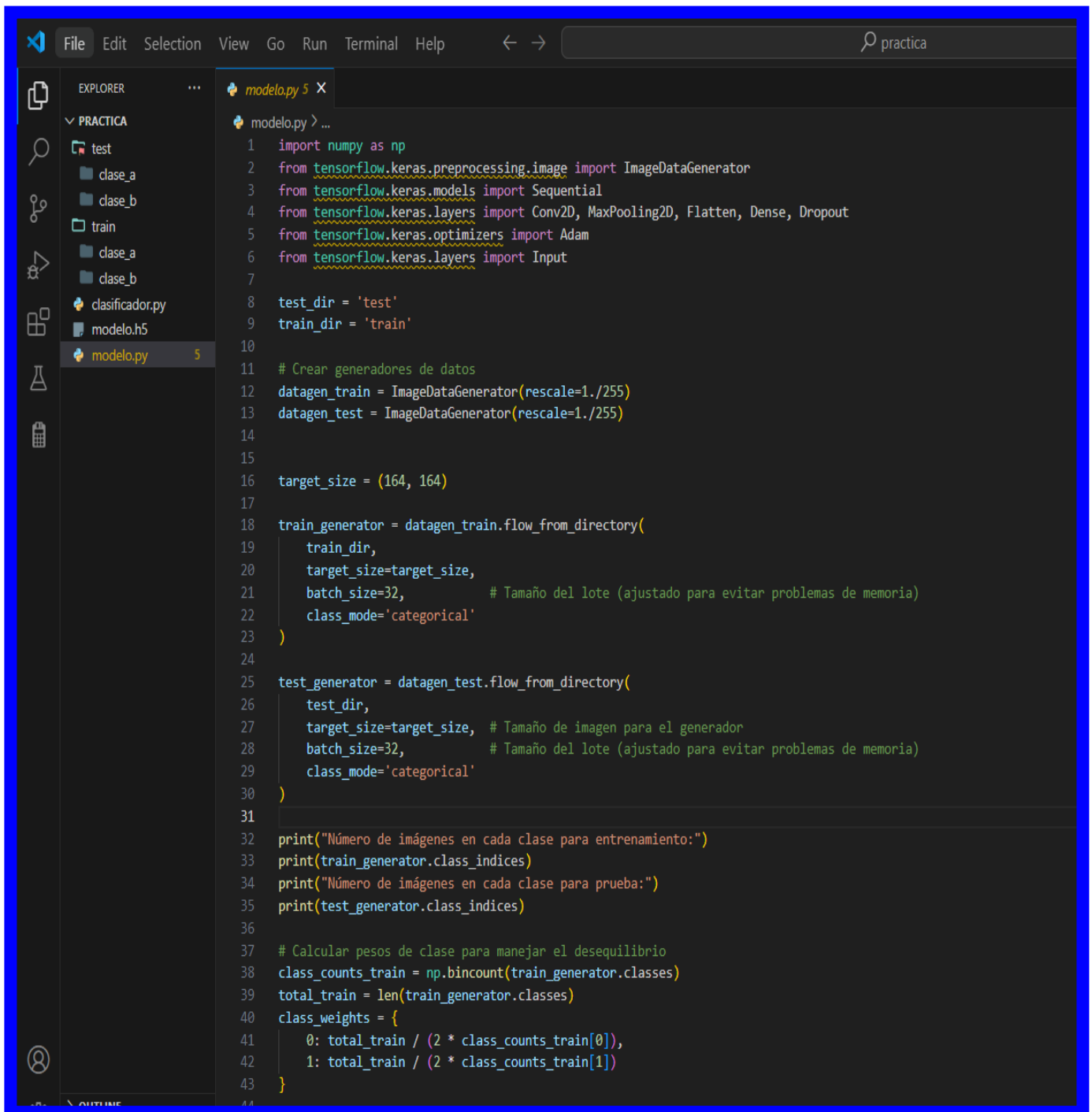
```
File Edit Selection View Go Run Terminal Help ← → practica

EXPLORER
PRACTICA
├── test
│   ├── clase_a
│   ├── clase_b
│   └── train
├── clasificador.py 2
├── modelo.h5
└── modelo.py

clasificador.py 2
1 model_path = modelo.h5
2 model = load_model(model_path)
3
4 def preprocesar_imagen(img_path, target_size=(164, 164)):
5     img = image.load_img(img_path, target_size=target_size)
6     img = image.img_to_array(img)
7     img = img / 255.0
8     img = img.reshape((1, 3, target_size[0], target_size[1]))
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

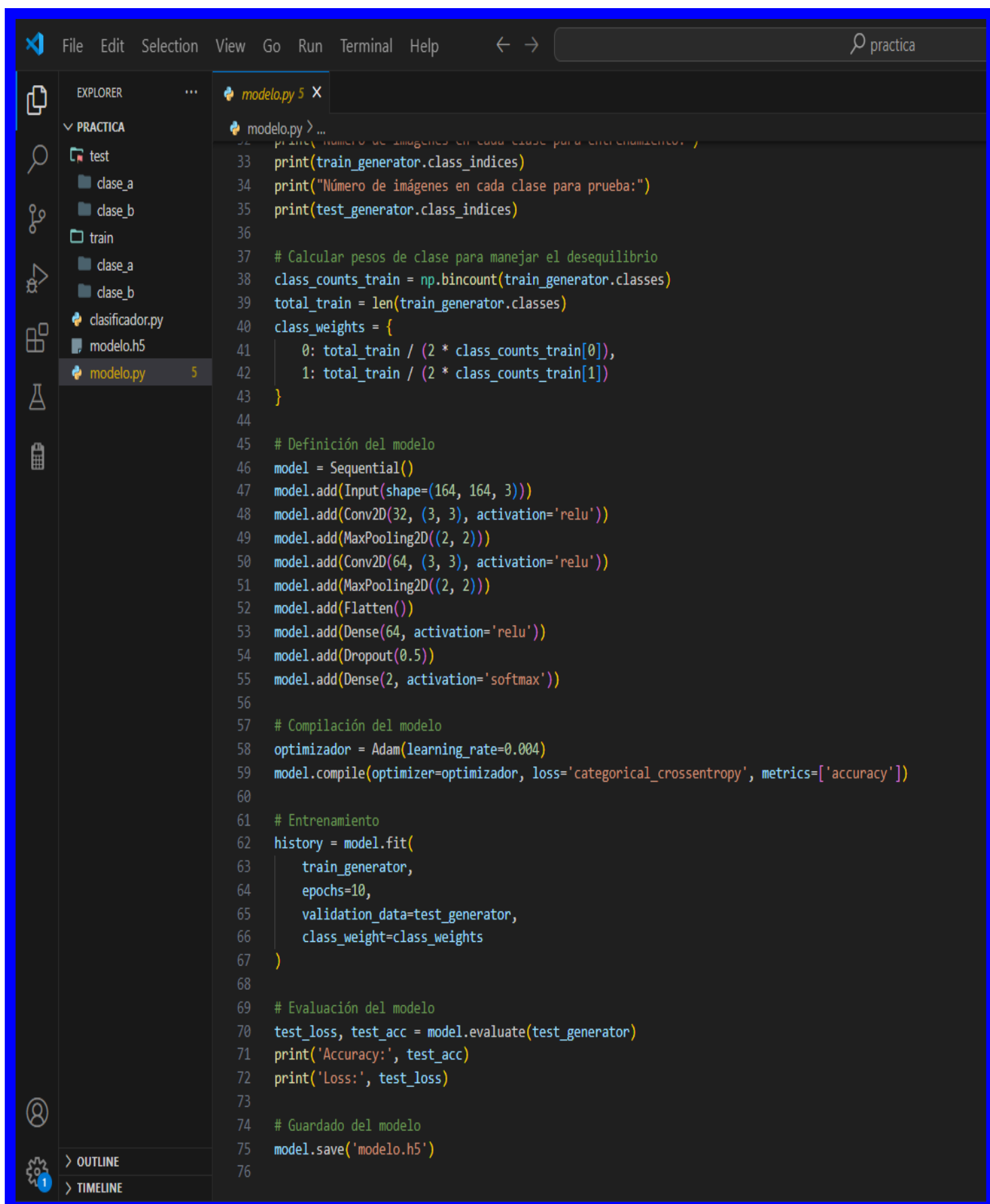
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS VIVOBOOK\Downloads\practica 1 (1)\practica> & "C:/Users/ASUS VIVOBOOK/AppData/Local/Programs/Python/Python312/python.exe" "C:/Users/ASUS VIVOBOOK/Do
2024-09-13 20:22:18.167267: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point
2024-09-13 20:22:22.475119: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point
2024-09-13 20:22:34.056282: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance
To enable the following instructions: AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. 'model.compile_metrics' will be empty until you train or evaluate the model.
1/1 1s 524ms/step
Salida predicha: [0.60129744 0.3987026 ]
Archivo: test/clase_a/4.PNG
Clase detectada: CLASE A
```



The image shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left shows a project named 'PRACTICA' with subfolders 'test' and 'train', each containing 'clase\_a' and 'clase\_b'. Files 'clasificador.py', 'modelo.h5', and 'modelo.py' are listed. 'modelo.py' is selected and open in the main editor. The code in 'modelo.py' is a Python script for image classification using TensorFlow and Keras. It imports necessary libraries, sets directory paths, creates data generators, and calculates class weights to handle class imbalance.

```
1  import numpy as np
2  from tensorflow.keras.preprocessing.image import ImageDataGenerator
3  from tensorflow.keras.models import Sequential
4  from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
5  from tensorflow.keras.optimizers import Adam
6  from tensorflow.keras.layers import Input
7
8  test_dir = 'test'
9  train_dir = 'train'
10
11 # Crear generadores de datos
12 datagen_train = ImageDataGenerator(rescale=1./255)
13 datagen_test = ImageDataGenerator(rescale=1./255)
14
15
16 target_size = (164, 164)
17
18 train_generator = datagen_train.flow_from_directory(
19     train_dir,
20     target_size=target_size,
21     batch_size=32,          # Tamaño del lote (ajustado para evitar problemas de memoria)
22     class_mode='categorical'
23 )
24
25 test_generator = datagen_test.flow_from_directory(
26     test_dir,
27     target_size=target_size, # Tamaño de imagen para el generador
28     batch_size=32,          # Tamaño del lote (ajustado para evitar problemas de memoria)
29     class_mode='categorical'
30 )
31
32 print("Número de imágenes en cada clase para entrenamiento:")
33 print(train_generator.class_indices)
34 print("Número de imágenes en cada clase para prueba:")
35 print(test_generator.class_indices)
36
37 # Calcular pesos de clase para manejar el desequilibrio
38 class_counts_train = np.bincount(train_generator.classes)
39 total_train = len(train_generator.classes)
40 class_weights = {
41     0: total_train / (2 * class_counts_train[0]),
42     1: total_train / (2 * class_counts_train[1])
43 }
```



```
32 print('Número de imágenes en cada clase para entrenamiento: ')
33 print(train_generator.class_indices)
34 print("Número de imágenes en cada clase para prueba:")
35 print(test_generator.class_indices)
36
37 # Calcular pesos de clase para manejar el desequilibrio
38 class_counts_train = np.bincount(train_generator.classes)
39 total_train = len(train_generator.classes)
40 class_weights = {
41     0: total_train / (2 * class_counts_train[0]),
42     1: total_train / (2 * class_counts_train[1])
43 }
44
45 # Definición del modelo
46 model = Sequential()
47 model.add(Input(shape=(164, 164, 3)))
48 model.add(Conv2D(32, (3, 3), activation='relu'))
49 model.add(MaxPooling2D((2, 2)))
50 model.add(Conv2D(64, (3, 3), activation='relu'))
51 model.add(MaxPooling2D((2, 2)))
52 model.add(Flatten())
53 model.add(Dense(64, activation='relu'))
54 model.add(Dropout(0.5))
55 model.add(Dense(2, activation='softmax'))
56
57 # Compilación del modelo
58 optimizador = Adam(learning_rate=0.004)
59 model.compile(optimizer=optimizador, loss='categorical_crossentropy', metrics=['accuracy'])
60
61 # Entrenamiento
62 history = model.fit(
63     train_generator,
64     epochs=10,
65     validation_data=test_generator,
66     class_weight=class_weights
67 )
68
69 # Evaluación del modelo
70 test_loss, test_acc = model.evaluate(test_generator)
71 print('Accuracy:', test_acc)
72 print('Loss:', test_loss)
73
74 # Guardado del modelo
75 model.save('modelo.h5')
76
```

EXPLORER

modelo.py X

PRACTICA

test

clase\_a

clase\_b

train

clasificador.py

modelo.h5

modelo.py 5

modelo.py &gt; ...

1 import numov as no

PROBLEMS 5

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

Epoch 2/10

2024-09-13 20:29:07.509540: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:29:07.510826: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 25s 12s/step - accuracy: 0.3803 - loss: 22.2035 - val\_accuracy: 0.3922 - val\_loss: 0.7822

Epoch 3/10

2024-09-13 20:29:32.362936: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:29:32.364744: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 26s 13s/step - accuracy: 0.5317 - loss: 0.6480 - val\_accuracy: 0.7843 - val\_loss: 0.5752

Epoch 4/10

2024-09-13 20:29:57.435195: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:29:57.436744: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 24s 12s/step - accuracy: 0.7994 - loss: 0.5805 - val\_accuracy: 0.8235 - val\_loss: 0.4164

Epoch 5/10

2024-09-13 20:30:21.217381: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:30:21.218721: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 24s 12s/step - accuracy: 0.8136 - loss: 0.4077 - val\_accuracy: 0.4706 - val\_loss: 1.0679

Epoch 6/10

2024-09-13 20:30:44.530823: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:30:44.531982: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 23s 11s/step - accuracy: 0.6378 - loss: 0.5969 - val\_accuracy: 0.8627 - val\_loss: 0.3129

Epoch 7/10

2024-09-13 20:31:07.247211: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:31:07.249569: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 23s 11s/step - accuracy: 0.8770 - loss: 0.3444 - val\_accuracy: 0.9608 - val\_loss: 0.2510

Epoch 8/10

2024-09-13 20:31:30.312259: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:31:30.315089: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 26s 14s/step - accuracy: 0.9470 - loss: 0.2571 - val\_accuracy: 0.9608 - val\_loss: 0.1527

Epoch 9/10

2024-09-13 20:31:56.285714: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:31:56.287694: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 23s 12s/step - accuracy: 0.9754 - loss: 0.1604 - val\_accuracy: 0.9412 - val\_loss: 0.1600

Epoch 10/10

2024-09-13 20:32:19.259117: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:450] ShuffleDatasetV3:2: Filling up shuffle buffer (this may take a while): 2 of 8

2024-09-13 20:32:19.260817: I tensorflow/core/kernels/data/shuffle\_dataset\_op.cc:480] Shuffle buffer filled.

2/2 ————— 23s 12s/step - accuracy: 0.8927 - loss: 0.2044 - val\_accuracy: 0.9216 - val\_loss: 0.3005

2/2 ————— 11s 5s/step - accuracy: 0.9165 - loss: 0.2737

Accuracy: 0.9215686321258545

Loss: 0.30051207542419434

**EVALUACIÓN DEL INFORME DE TRABAJO SEMANAL**

NOTA

**OBSERVACIONES Y RECOMENDACIONES**

DEL INSTRUCTOR:


FIRMA DEL ESTUDIANTE:

FIRMA DEL INSTRUCTOR:



**PROPIEDAD INTELECTUAL DEL SENATI. PROHIBIDA SU  
REPRODUCCIÓN Y VENTA SIN LA AUTORIZACIÓN  
CORRESPONDIENTE**