

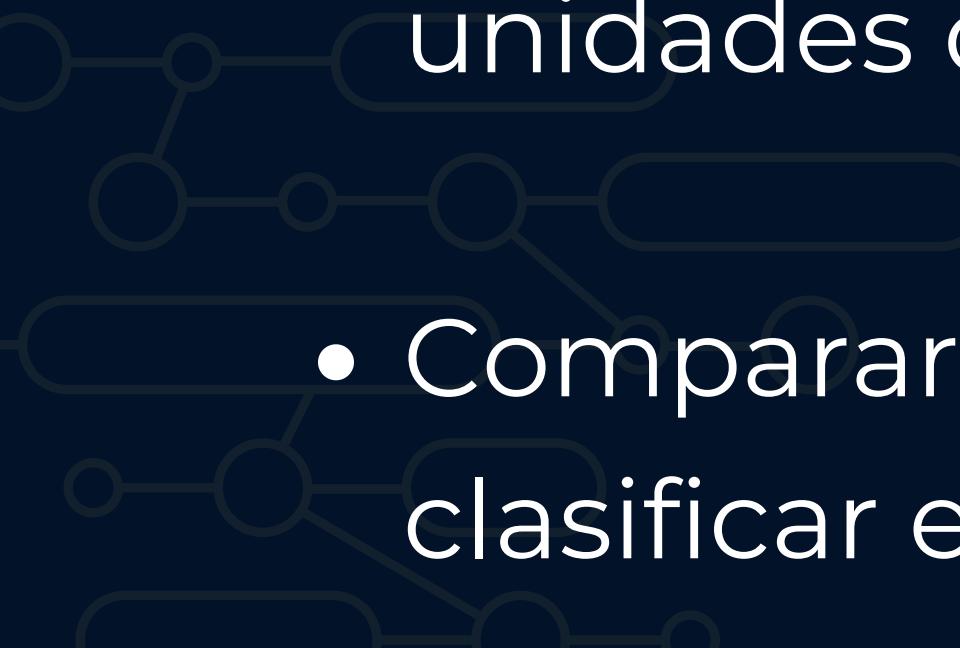
HEALTHY OR UNHEALTHY

David Delgado - A01641526



Colaboradores: Hugo Aguayo y
AI colab

FUNCIÓN

- Extraer texto de una etiqueta con OCR (Tesseract).
 - Detectar nutrientes y unidades con regex.
- 
- Comparar con umbrales y clasificar el alimento.

Declaración Nutrimental	
CONTENIDO ENERGÉTICO POR ENVASE	1 656 kcal (6 888 kJ)
CONTENIDO ENERGÉTICO POR 100 g	138 kcal (574 kJ)
PROTEÍNAS	12 g
GRASAS TOTALES	10 g
GRASAS SATURADAS	3 g
GRASAS TRANS	0 mg
HIDRATOS DE CARBONO DISPONIBLES	0 g
AZÚCARES	0 g
AZÚCARES AÑADIDOS	0 g
FIBRA DIETÉTICA	4 g
SODIO	130 mg

--- Clasificación del Alimento ---
Unhealthy

PIPELINE

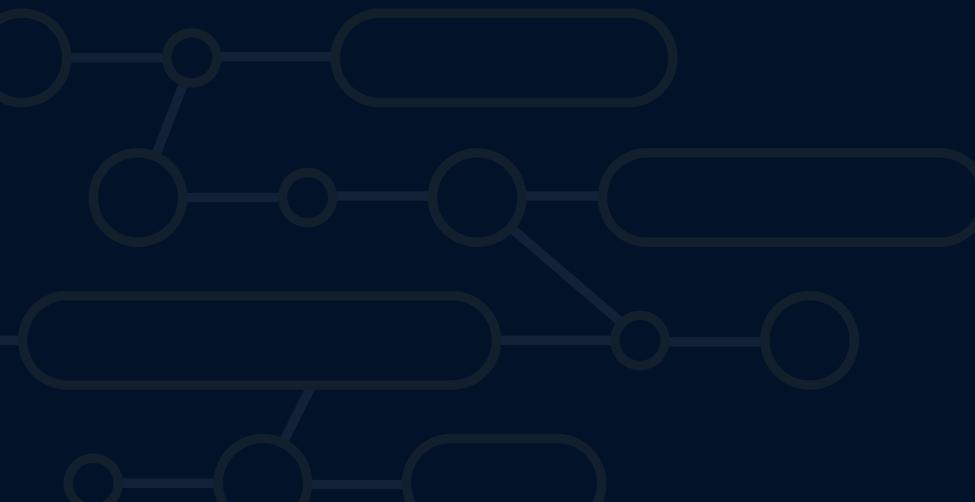
- *Cargar imagen y convertir BGR→RGB.*
- *OCR: pyt.image_to_string.*
- *Parseo línea a línea con regex.*
- *Diccionario nutrient_info.*
- *Reglas → clasificación final.*

```
import pytesseract as pyt
import cv2
import numpy as np
import matplotlib.pyplot as plt
etiqueta = cv2.imread("etiquetado3.png")
etiqueta = cv2.cvtColor(etiqueta, cv2.COLOR_BGR2RGB)
plt.imshow(etiqueta)
plt.axis("off")
extracted_text = pyt.image_to_string(etiqueta)

print("Extracted test:", extracted_text)
```

OCR EN 2 LÍNEAS

- etiqueta = cv2.imread(...) +
cvtColor(...).
- extracted_text =
pyt.image_to_string(etiqueta).



PARSEO CON REGEX

```
import re

nutrient_info = {}
lines = extracted_text.splitlines()

for line in lines:
    # Use regex to find nutrient names and values (handling potential variations)
    match = re.match(r'([A-Z\s]+)\s+(\d+)([gk]mgkcal]+)', line, re.IGNORECASE)
    if match:
        nutrient_name = match.group(1).strip()
        nutrient_value = match.group(2)
        nutrient_unit = match.group(3)
        nutrient_info[nutrient_name] = f"{nutrient_value}{nutrient_unit}"
```

```
else:
    if "PROTEINAS" in line:
        match = re.search(r'PROTEINAS\s*[:\s]*\s*(\d+(\.\d+)?)\s*g', line, re.IGNORECASE)
        if match:
            nutrient_info["PROTEINAS"] = match.group(1) + "g"
if "PROTEINAS" in line:
    match = re.search(r'PROTEINAS\s*[:\s]*\s*(\d+(\.\d+)?)\s*g', line, re.IGNORECASE)
    if match:
        nutrient_info["PROTEINAS"] = match.group(1) + "g"
```

- Patrón general: NOMBRE — VALOR — UNIDAD.
- Soporta nombres con espacios y unidad opcional.
- Casos especiales (p. ej. “SODIO”, “GRASAS TRANS”).

REGLAS DE CLASIFICACIÓN

Diccionario rules con umbral + comparador.

01

Unhealthy (si
un exceso)

02

Healthy (si
todo cumple)

03

Low in ... (si
aplica)

04

No
clasificable.

```
rules = {
    "Unhealthy": {
        "GRASAS TOTALES": {"threshold": 17.5, "comparison": ">"},
        "GRASAS SATURADAS": {"threshold": 5, "comparison": ">"},
        "AZUCARES": {"threshold": 22.5, "comparison": ">"},
        "SODIO": {"threshold": 600, "comparison": ">"}}
```

```
is_unhealthy = False
for rule, conditions in rules["Unhealthy"].items():
    if rule in nutrient_values_float and nutrient_values_float[rule] is not None:
        if conditions["comparison"] == ">" and nutrient_values_float[rule] > conditions["threshold"]:
            is_unhealthy = True
            break
```

```
if is_low_in_fat:
    food_classification = "Low in Fat"
elif is_low_in_saturated_fat:
    food_classification = "Low in Saturated Fat"
elif is_low_in_sugar:
    food_classification = "Low in Sugar"
elif is_low_in_sodium:
    food_classification = "Low in Sodium"
else:
    food_classification = "Cannot be classified based on available information"
```

EJEMPLO DE SALIDA

- Información Nutricional (diccionario impreso).
- Clasificación final: “Unhealthy.

--- Información Nutricional ---

CONTENIDO ENERGETICO: 138kcal

POR: 100g

PROTEINAS: 12g

GRASAS TOTALES: 10g

GRASAS SATURADAS: 3g

HIDRATOS DE CARBONO DISPONIBLES: 0g

--- Clasificación del Alimento ---

Unhealthy

Limitaciones

- OCR puede fallar con baja resolución o contrastes.
- Nombres con acentos/variantes.
- Unidades: mg↔g (normalizar para más precisión).
- capturar %VD, “por porción” vs “por 100 g”, y multi-idioma.

Conclusiones

- OCR logra medianamente el propósito
- Determina si es saludable o no
- No funciona al 100%

GRACIAS

