import rasterio

import numpy as np

from sklearn.ensemble import RandomForestClassifier

from sklearn.model\_selection import train\_test\_split

import matplotlib.pyplot as plt

# 1. Cargar imagen satelital

ruta\_imagen = 'tu\_imagen.tif' # Cambia esto a la ruta de tu imagen

with rasterio.open(ruta\_imagen) as src:

imagen = src.read() # (bandas, filas, columnas)

perfil = src.profile

# 2. Preparar los datos

# Convertimos la imagen en una matriz de píxeles

n\_bandas, n\_filas, n\_columnas = imagen.shape

X = imagen.reshape(n\_bandas, -1).T # (n\_pixels, n\_bandas)

# Opcional: eliminar datos inválidos (nubes, sombras, bordes)

# Aquí asumimos que 0 en todas las bandas es no válido

mascara\_valida = np.all(X > 0, axis=1)

X\_validos = X[mascara\_valida]

# 3. Crear etiquetas de entrenamiento (esto normalmente se crea a mano)

# Para este ejemplo, generamos etiquetas falsas

# En un caso real usarías puntos de muestreo conocidos

n\_muestras = 5000

indices\_muestra = np.random.choice(X\_validos.shape[0], n\_muestras, replace=False)

X\_train = X\_validos[indices\_muestra]

y\_train = np.random.randint(0, 3, n\_muestras) # Por ejemplo 3 clases

# 4. Entrenar clasificador

modelo = RandomForestClassifier(n\_estimators=100, random\_state=42)

modelo.fit(X\_train, y\_train)

# 5. Clasificar toda la imagen

y\_pred = np.full(X.shape[0], -1) # Inicializamos todo como "no clasificado"

y\_pred[mascara\_valida] = modelo.predict(X\_validos)

# 6. Volver a la forma espacial

clasificacion = y\_pred.reshape(n\_filas, n\_columnas)

# 7. Mostrar resultado

plt.imshow(clasificacion, cmap='viridis')

plt.title('Clasificación de Imagen Satelital')

plt.colorbar()

plt.show()

# 8. Guardar la clasificación

perfil.update(dtype=rasterio.int16, count=1)

with rasterio.open('clasificacion.tif', 'w', \*\*perfil) as dst:

dst.write(clasificacion.astype(rasterio.int16), 1)