from PIL import Image

import numpy as np

import matplotlib.pyplot as plt

def skewness(data):

n = len(data)

mean = np.mean(data)

std\_dev = np.std(data)

skew = np.sum(((data - mean) / std\_dev) \*\* 3) / n

return skew

def kurtosis(data):

n = len(data)

mean = np.mean(data)

std\_dev = np.std(data)

kurt = np.sum(((data - mean) / std\_dev) \*\* 4) / n - 3 # Excess kurtosis

return kurt

def binary\_image\_statistics(image\_path):

# Load and convert image to grayscale

image = Image.open(image\_path).convert('L')

image\_array = np.array(image)

# Apply simple thresholding (e.g., Otsu's can be added if needed manually)

threshold = 128

binary\_image = (image\_array > threshold) \* 255

binary\_image = binary\_image.astype(np.uint8)

# Flatten image to 1D array

pixel\_values = binary\_image.ravel()

# Compute statistics

stats\_values = {

'Mean': np.mean(pixel\_values),

'Median': np.median(pixel\_values),

'Standard Deviation': np.std(pixel\_values),

'Skewness': skewness(pixel\_values),

'Kurtosis': kurtosis(pixel\_values)

}

# Display the binary image

plt.imshow(binary\_image, cmap='gray')

plt.axis('off')

plt.title('Binary Image')

plt.show()

# Print statistics

print("Statistics values for binary image:")

for stat, value in stats\_values.items():

print(f"{stat}: {value:.2f}")

return stats\_values

# Example usage

image\_path = r"C:/Users/gaura/Downloads/flower.png" # Replace with your image path

stats\_values = binary\_image\_statistics(image\_path)