In [3]:
```
# CODE FOR EXTRACTING ALL 34 FEATURES OF TRAIN(RAVELING & NON-RAVELING) FOLDER o
# This is the code for the extraction of features of Raveling images, I extracte
# Then both csv files are merged containing features of both named as 'images_fo
# importing required libraries for extracting 2 types of features color-based an
import os  #For interacting with the operating system, such as file and director
import cv2 # OpenCV library for computer vision tasks,image reading and processi
import mahotas # Library for image processing(for extracting texture features (G
import numpy as np
from scipy.stats import skew, kurtosis, entropy # for statistical analysis
import pandas as pd

# Defining a function to calculate entropy manually using histogram
def calculate_entropy(channel):
    """Calculate entropy of an image channel using histogram."""
    hist, _ = np.histogram(channel, bins=256, range=(0, 256), density=True)
    return entropy(hist, base=2)  # Base 2 entropy

# Defining a function to extract color-based features of each channel Red,Green,
def extract_color_features(image):
    features = {}
    # Split into color channels
    blue, green, red = cv2.split(image)

    # Calculating the mean, std, skewness, kurtosis, entropy, and range for each
    for channel_name, channel in zip(['red', 'green', 'blue'], [red, green, blue
        features[f'mean_{channel_name}'] = np.mean(channel)
        features[f'std_{channel_name}'] = np.std(channel)
        features[f'skewness_{channel_name}'] = skew(channel.flatten())
        features[f'kurtosis_{channel_name}'] = kurtosis(channel.flatten())
        features[f'entropy_{channel_name}'] = calculate_entropy(channel)
        features[f'range_{channel_name}'] = np.ptp(channel)  # Peak-to-peak rang

    return features

# Defining a function to extract GLCM texture-based features using mahotas
def extract_glcm_features(image):
    features = {}
    # Converting the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Calculate GLCM using mahotas
    glcm = mahotas.features.haralick(gray_image)

    # Extract specific features for each direction (0°, 45°, 90°, 135°) total of
    angles = ['0_deg', '45_deg', '90_deg', '135_deg']
    for i, angle in enumerate(angles):
        features[f'contrast_{angle}'] = glcm[i, 1]   # Contrast
        features[f'correlation_{angle}'] = glcm[i, 2]   # Correlation
        features[f'asm_{angle}'] = glcm[i, 0]   # Angular Second Moment (ASM)
        features[f'entropy_{angle}'] = glcm[i, 8]   # Entropy

    return features

# Defining a function to extract all features (color and GLCM) from an image
def extract_all_features(image):
    features = {}
    features.update(extract_color_features(image))
    features.update(extract_glcm_features(image))
```

```
    return features

# Folder where images are stored
image_folder_path = "C:\Raveling"

# List all files in the folder and filter only .jpg images
image_files = [f for f in os.listdir(image_folder_path) if f.endswith('.jpg')]

# Initialize an empty list to store features for each image
features_list = []

# Loop through each image and extract features
for image_file in image_files:
    image_path = os.path.join(image_folder_path, image_file)
    image = cv2.imread(image_path)

    if image is not None:
        # Extract features from the image
        features = extract_all_features(image)
        features['image_name'] = image_file  # Include the image file name for r
        features_list.append(features)

# Convert the list of features into a DataFrame
features_df = pd.DataFrame(features_list)

# Save the extracted features to a CSV file
csv_file_path = 'raveling_image_features.csv'
features_df.to_csv(csv_file_path, index=False)

print(f"Feature extraction complete. CSV saved at {csv_file_path}")
```

```
<>:61: SyntaxWarning: invalid escape sequence '\R'
<>:61: SyntaxWarning: invalid escape sequence '\R'
C:\Users\HP\AppData\Local\Temp\ipykernel_8008\2141231535.py:61: SyntaxWarning: in
valid escape sequence '\R'
  image_folder_path = "C:\Raveling"
Feature extraction complete. CSV saved at raveling_image_features.csv
```

In [7]:

```
# CODE FOR EXTRACTING ALL 34 FEATURES OF TEST(RAVELING & NON-RAVELING) FOLDER OF
import os
import cv2
import mahotas
import numpy as np
from scipy.stats import skew, kurtosis, entropy
import pandas as pd

# Define a function to calculate entropy manually using histogram
def calculate_entropy(channel):
    """Calculate entropy of an image channel using histogram."""
    hist, _ = np.histogram(channel, bins=256, range=(0, 256), density=True)
    return entropy(hist, base=2)  # Base 2 entropy

# Define a function to extract color-based features
def extract_color_features(image):
    features = {}
    # Split into color channels
    blue, green, red = cv2.split(image)

    # Calculate the mean, std, skewness, kurtosis, entropy, and range for each c
    for channel_name, channel in zip(['red', 'green', 'blue'], [red, green, blue
```

```
        features[f'mean_{channel_name}'] = np.mean(channel)
        features[f'std_{channel_name}'] = np.std(channel)
        features[f'skewness_{channel_name}'] = skew(channel.flatten())
        features[f'kurtosis_{channel_name}'] = kurtosis(channel.flatten())
        features[f'entropy_{channel_name}'] = calculate_entropy(channel)
        features[f'range_{channel_name}'] = np.ptp(channel)  # Peak-to-peak rang

    return features

# Define a function to extract GLCM texture-based features using mahotas
def extract_glcm_features(image):
    features = {}
    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Calculate GLCM using mahotas
    glcm = mahotas.features.haralick(gray_image)

    # Extract specific features for each direction (0°, 45°, 90°, 135°)
    angles = ['0_deg', '45_deg', '90_deg', '135_deg']
    for i, angle in enumerate(angles):
        features[f'contrast_{angle}'] = glcm[i, 1]   # Contrast
        features[f'correlation_{angle}'] = glcm[i, 2]   # Correlation
        features[f'asm_{angle}'] = glcm[i, 0]   # Angular Second Moment (ASM)
        features[f'entropy_{angle}'] = glcm[i, 8]   # Entropy

    return features

# Define a function to extract all features (color and GLCM) from an image
def extract_all_features(image):
    features = {}
    features.update(extract_color_features(image))
    features.update(extract_glcm_features(image))
    return features

# Folder where images are stored
image_folder_path = "C:\Raveling_non_raveling"

# List all files in the folder and filter only .jpg images
image_files = [f for f in os.listdir(image_folder_path) if f.endswith('.jpg')]

# Initialize an empty list to store features for each image
features_list = []

# Loop through each image and extract features
for image_file in image_files:
    image_path = os.path.join(image_folder_path, image_file)
    image = cv2.imread(image_path)

    if image is not None:
        # Extract features from the image
        features = extract_all_features(image)
        features['image_name'] = image_file  # Include the image file name for r
        features_list.append(features)

# Convert the list of features into a DataFrame
features_df = pd.DataFrame(features_list)

# Save the extracted features to a CSV file
csv_file_path = 'test_images_features_.csv'
```

```
features_df.to_csv(csv_file_path, index=False)

print(f"Feature extraction complete. CSV saved at {csv_file_path}")
```

```
<>:59: SyntaxWarning: invalid escape sequence '\R'
<>:59: SyntaxWarning: invalid escape sequence '\R'
C:\Users\HP\AppData\Local\Temp\ipykernel_8008\1965315743.py:59: SyntaxWarning: in
valid escape sequence '\R'
  image_folder_path = "C:\Raveling_non_raveling"
Feature extraction complete. CSV saved at test_images_features_.csv
```

In [ ]: