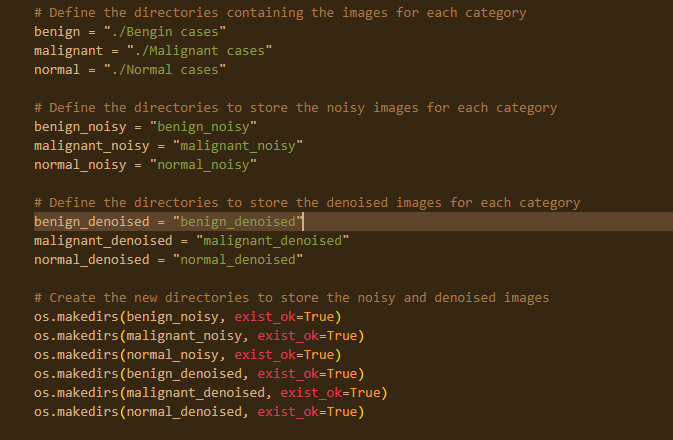
**Image Denoising Report**

This report outlines the process of image denoising using various techniques and provides visualizations of the denoised images. The code snippet provided demonstrates the denoising process for a specific category of images, specifically benign cases. Here are the key steps involved:

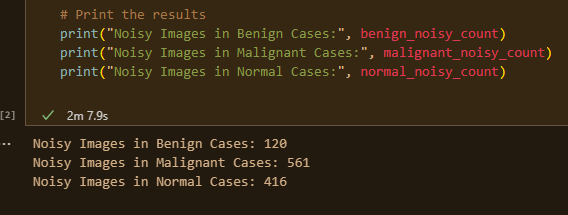
**Directory Definitions**

Directories are defined to store the original images, noisy images, and denoised images for each category: benign, malignant, and normal. Separate directories are created to store the noisy and denoised images.



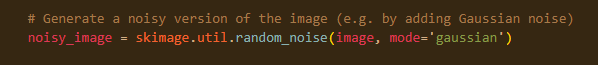
**Find the number of noisy images per directory**

The mean squared error and the peak signal noise ratio are used to determine if an image is noisy. The following screenshot shows the total number of noisy images per directory

****

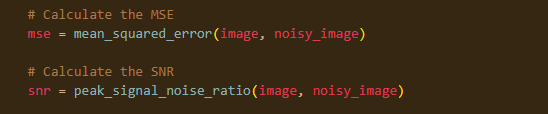
**Generating Noisy Image**

A noisy version of the image is generated by adding Gaussian noise using the skimage.util.random\_noise() function.



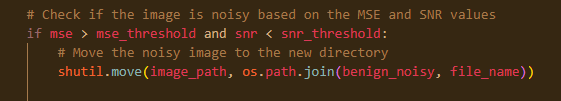
**Calculating MSE and SNR**

The Mean Squared Error (MSE) and Signal-to-Noise Ratio (SNR) are calculated to evaluate the noise level in the image. These metrics are computed using the skimage.metrics.mean\_squared\_error() and skimage.metrics.peak\_signal\_noise\_ratio() functions, respectively.



**Checking Noise Level**

The MSE and SNR values are used to determine whether the image is considered noisy. If the MSE is above the defined threshold and the SNR is below the defined threshold, the image is considered noisy.

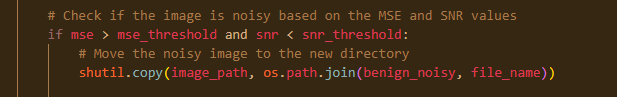


**Handling Noisy Images**

For noisy images, the following steps are performed:

**Moving the Noisy Image**

The noisy image is moved to the corresponding directory for storing noisy images using the shutil.copy() function.



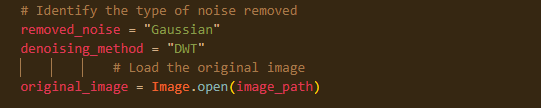
**Denoising the Image with DWT**

The noisy image is denoised using the Discrete Wavelet Transform (DWT) technique. The DWT coefficients are calculated using the pywt.dwt2() function, and denoising is performed using the denoise\_wavelet() function. The denoised image is reconstructed using the inverse DWT (pywt.idwt2()) and then clipped to the valid pixel value range. The resulting denoised image is saved as an 8-bit grayscale image using the skimage.io.imsave() function.



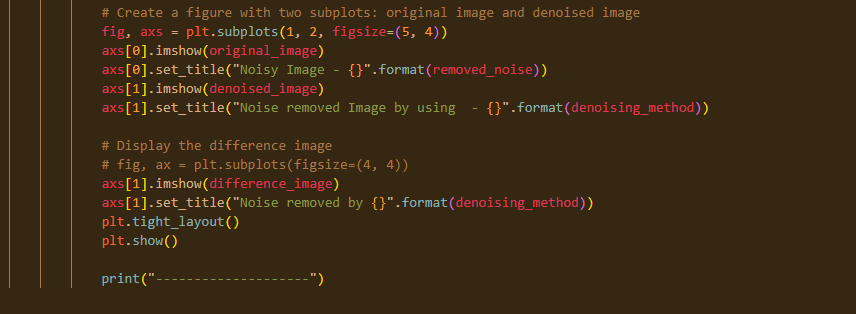
**Identifying Noise Type and Denoising Method**

The type of noise removed (in this case, Gaussian noise) and the denoising method (DWT) are identified.



**Displaying Results**

Various visualizations and information are displayed for the denoised image, including the noise density, the original image with noise, the denoised image, and the difference image between the original and denoised versions.



**Visualization**

To provide visualizations of the denoised images and the difference between the original and denoised images, the following steps are performed:

The original image with noise and the denoised image are displayed side by side in a figure using the matplotlib.pyplot.imshow() function.

The difference image between the original and denoised images is calculated using ImageChops.difference().

The difference image and denoised image are displayed in a subplot for visual comparison.

