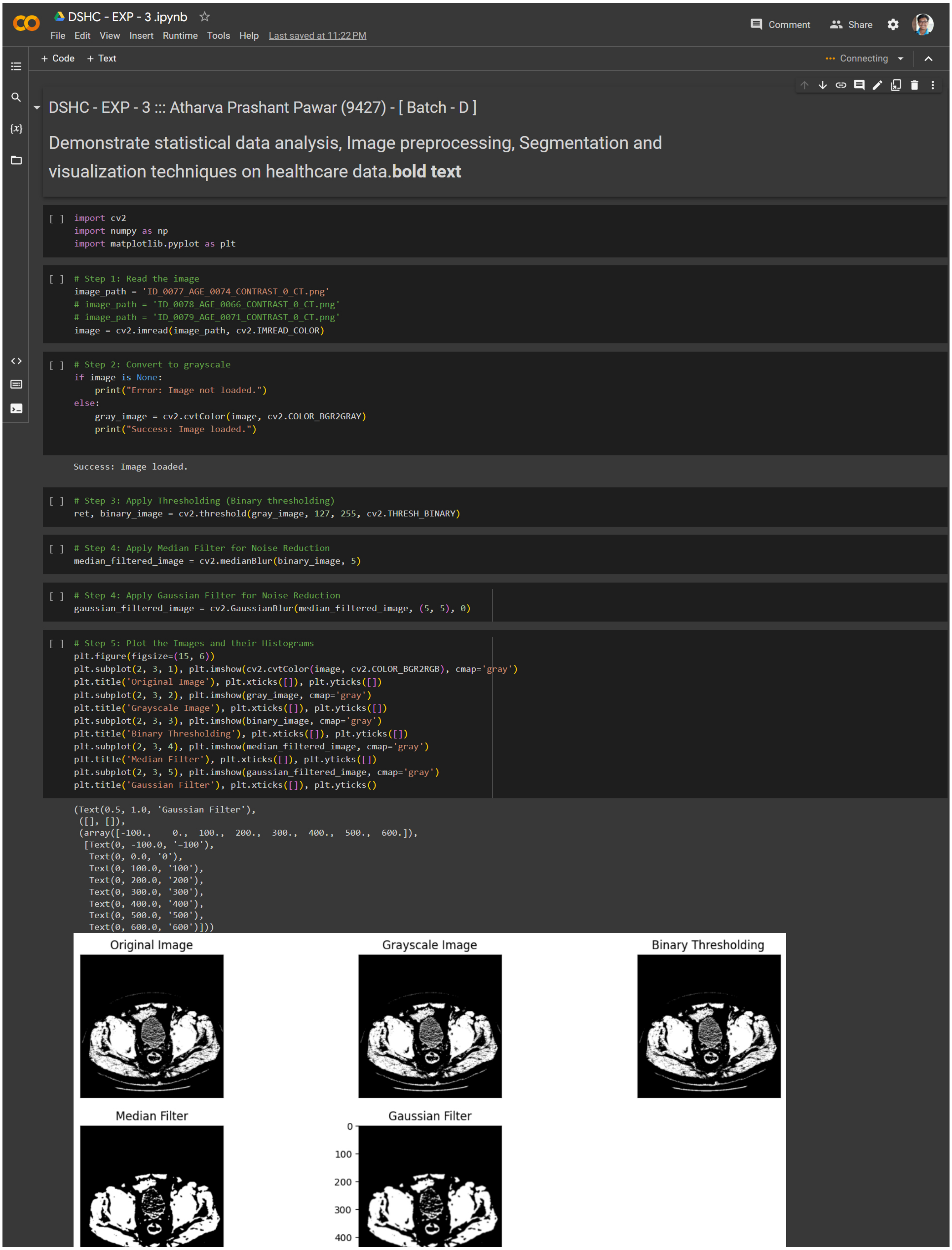


DSHC - EXP - 3 .ipynb

File Edit View Insert Runtime Tools Help Last saved at 11:22 PM

Comment Share

+ Code + Text

Connecting

DSHC - EXP - 3 ::: Atharva Prashant Pawar (9427) - [Batch - D]

Demonstrate statistical data analysis, Image preprocessing, Segmentation and visualization techniques on healthcare data.**bold text**

```
[ ] import cv2
    import numpy as np
    import matplotlib.pyplot as plt
```

```
[ ] # Step 1: Read the image
    image_path = 'ID_0077_AGE_0074_CONTRAST_0_CT.png'
    # image_path = 'ID_0078_AGE_0066_CONTRAST_0_CT.png'
    # image_path = 'ID_0079_AGE_0071_CONTRAST_0_CT.png'
    image = cv2.imread(image_path, cv2.IMREAD_COLOR)
```

```
[ ] # Step 2: Convert to grayscale
    if image is None:
        print("Error: Image not loaded.")
    else:
        gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        print("Success: Image loaded.")
```

```
    Success: Image loaded.
```

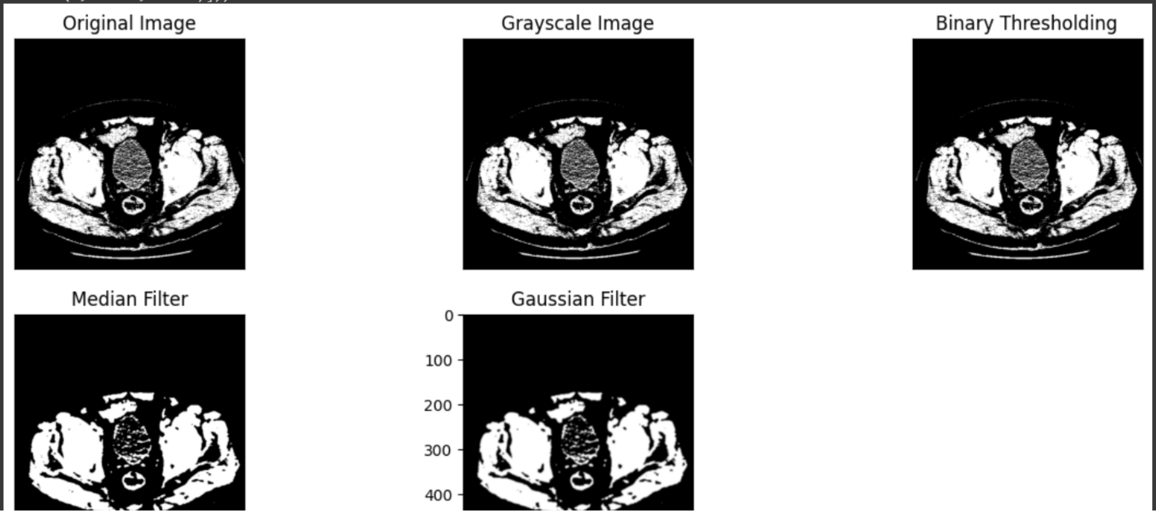
```
[ ] # Step 3: Apply Thresholding (Binary thresholding)
    ret, binary_image = cv2.threshold(gray_image, 127, 255, cv2.THRESH_BINARY)
```

```
[ ] # Step 4: Apply Median Filter for Noise Reduction
    median_filtered_image = cv2.medianBlur(binary_image, 5)
```

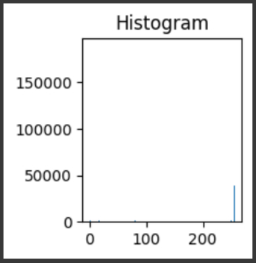
```
[ ] # Step 4: Apply Gaussian Filter for Noise Reduction
    gaussian_filtered_image = cv2.GaussianBlur(median_filtered_image, (5, 5), 0)
```

```
[ ] # Step 5: Plot the Images and their Histograms
    plt.figure(figsize=(15, 6))
    plt.subplot(2, 3, 1), plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB), cmap='gray')
    plt.title('Original Image'), plt.xticks([]), plt.yticks([])
    plt.subplot(2, 3, 2), plt.imshow(gray_image, cmap='gray')
    plt.title('Grayscale Image'), plt.xticks([]), plt.yticks([])
    plt.subplot(2, 3, 3), plt.imshow(binary_image, cmap='gray')
    plt.title('Binary Thresholding'), plt.xticks([]), plt.yticks([])
    plt.subplot(2, 3, 4), plt.imshow(median_filtered_image, cmap='gray')
    plt.title('Median Filter'), plt.xticks([]), plt.yticks([])
    plt.subplot(2, 3, 5), plt.imshow(gaussian_filtered_image, cmap='gray')
    plt.title('Gaussian Filter'), plt.xticks([]), plt.yticks()
```

```
    (Text(0.5, 1.0, 'Gaussian Filter'),
     ([], []),
     (array([-100.,    0.,  100.,  200.,  300.,  400.,  500.,  600.]),
      [Text(0, -100.0, '-100'),
       Text(0, 0.0, '0'),
       Text(0, 100.0, '100'),
       Text(0, 200.0, '200'),
       Text(0, 300.0, '300'),
       Text(0, 400.0, '400'),
       Text(0, 500.0, '500'),
       Text(0, 600.0, '600')]))
```

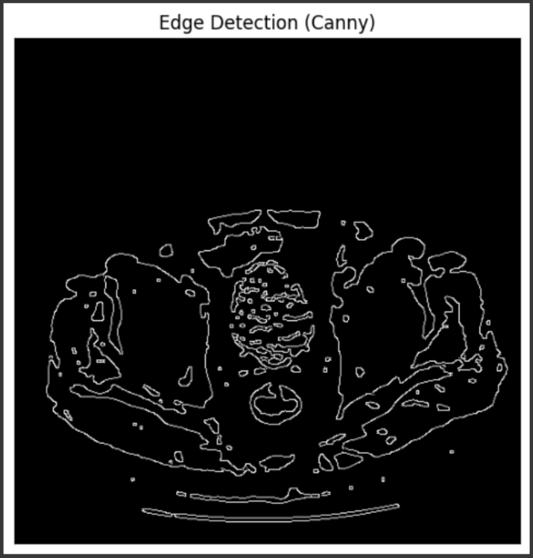


```
[20] # Plot histograms
     plt.subplot(2, 3, 6), plt.hist(gaussian_filtered_image.ravel(), 256, [0, 256])
     plt.title('Histogram')
     plt.show()
```



```
[21] # Step 6: Detection of Edges (Canny Edge Detection)
     edges = cv2.Canny(gaussian_filtered_image, 100, 200)
```

```
[22] # Display the edges
     plt.figure(figsize=(6, 6))
     plt.imshow(edges, cmap='gray')
     plt.title('Edge Detection (Canny)')
     plt.xticks([]), plt.yticks([])
     plt.show()
```

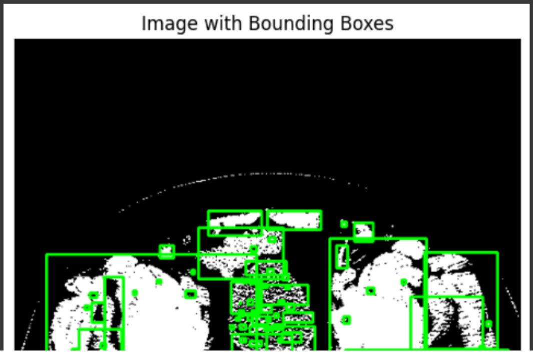


```
[23] # Find contours in the edge image
     contours, _ = cv2.findContours(edges, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

     # Create a copy of the original image to draw bounding boxes on
     image_with_boxes = image.copy()

     # Iterate through the contours and draw bounding boxes
     for contour in contours:
         x, y, w, h = cv2.boundingRect(contour)
         cv2.rectangle(image_with_boxes, (x, y), (x + w, y + h), (0, 255, 0), 2)  # Green rectangle

     # Display the image with bounding boxes
     plt.figure(figsize=(6, 6))
     plt.imshow(cv2.cvtColor(image_with_boxes, cv2.COLOR_BGR2RGB))
     plt.title('Image with Bounding Boxes')
     plt.xticks([]), plt.yticks([])
     plt.show()
```



```
[ ]
```

Colab paid products - Cancel contracts here

Connecting to Python 3 Google Compute Engine backend