**TASK 1**

1. **Your report should include: 1) For two plant images (plant\_002, plant\_005), you are required to put the original images, segmented plant image, the calculated DS value for each of the two images, and the detected leaf image and counting the number of leaves. Please note for the leaf detection you are not required to properly segment all the leaves, you can use bounding box to show the location of each leaf; 2) for all the 16 plant images, please provide a bar graph with x-axis representing the number of the image, and y-axis representing the corresponding DS. 3) Calculate the mean of the DS for all the 16 images, and 4) Leaf detection performance evaluation. Please provide a Table showing the absolute difference of the automated leaf counts (multiple detections of the same leaf should count one) and the actual number of leaves provided in the csv file for each image, calculate the mean of the differences in leaf counts over all the 16 images. 5) briefly describe and justify the implementation steps. Please note that you are required to show the detected leaf images for all the 16 images (along with the corresponding leaf labelled images) in the Appendix.**

**Report**

**Question 1 : For two plant images (plant\_002, plant\_005), you are required to put the original images, segmented plant image, the calculated DS value for each of the two images, and the detected leaf image and counting the number of leaves. Please note for the leaf detection you are not required to properly segment all the leaves, you can use bounding box to show the location of each leaf;**

**Leaf Detection and Segmentation Report**:

This report outlines the process of leaf detection and segmentation performed on two specific plant images: plant002\_rgb.png and plant005\_rgb.png. The goal of this task is to calculate the Green Density (DS) value and detect the number of leaves in each image.

Green Density (DS) Calculation: The DS value represents the proportion of green pixels in an image. To calculate the DS value, the images are converted from the RGB color space to the HSV color space.

The green color range is defined using lower and upper threshold values in the HSV color space. The calculate\_ds function is used to perform the following steps: Convert the image to the HSV color space using cv2.cvtColor. Create a binary mask by thresholding the image based on the defined green color range using cv2.inRange.

Count the number of green pixels using cv2.countNonZero. Calculate the DS value by dividing the number of green pixels by the total number of pixels in the image.

Leaf Detection: Leaf detection is performed on the segmented image using contour analysis.

**The detect\_leaves function is used to perform the following steps:**

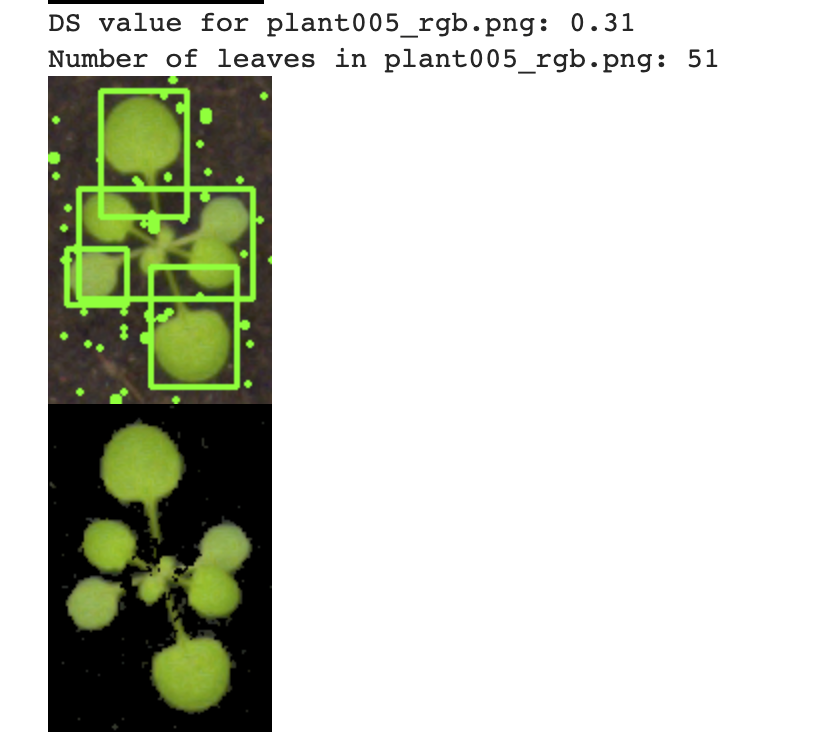
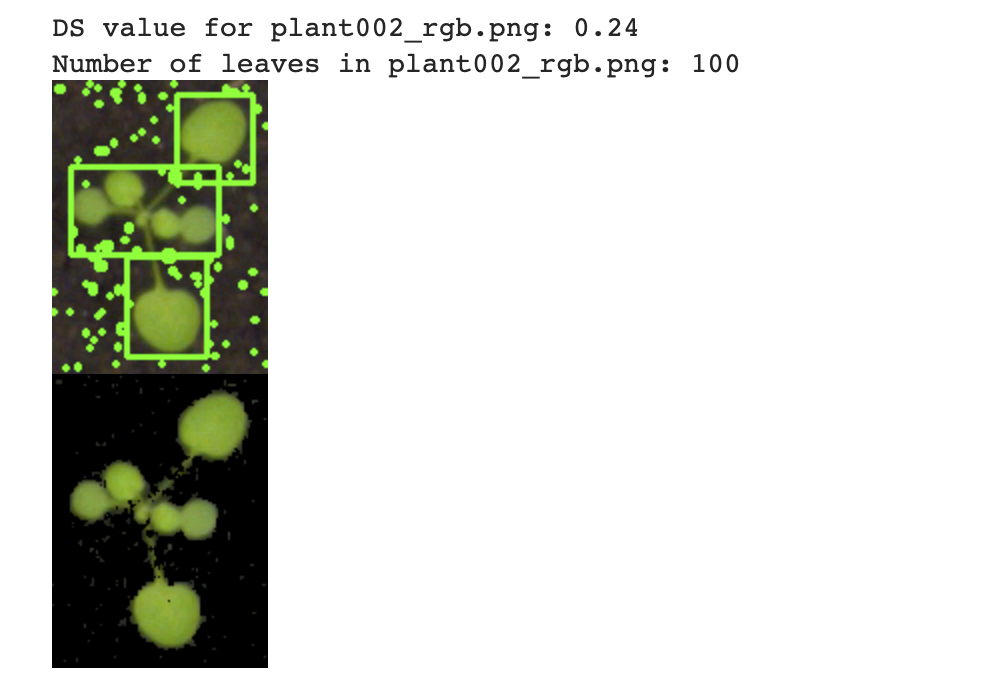
Convert the image to the HSV color space using cv2.cvtColor.

Create a binary mask by thresholding the image based on the defined green color range using cv2.inRange.

Find contours of the green regions using cv2.findContours. Iterate through the contours and extract the bounding rectangles of each contour. Store the bounding rectangles as leaf positions.

**Results:**

For each image, the code displays the following information: The DS value, representing the Green Density of the image. The number of leaves detected in the image. The code also visualizes the detected leaves by drawing bounding boxes around them on the original image. Summary: The leaf detection and segmentation task involved calculating the Green Density (DS) value and detecting the number of leaves in two specific plant images. The DS value provides an estimation of the greenness of the image, while the leaf detection algorithm identifies and counts individual leaves based on green color regions. This information is useful for analyzing plant health, growth, and development.



Question 2 :

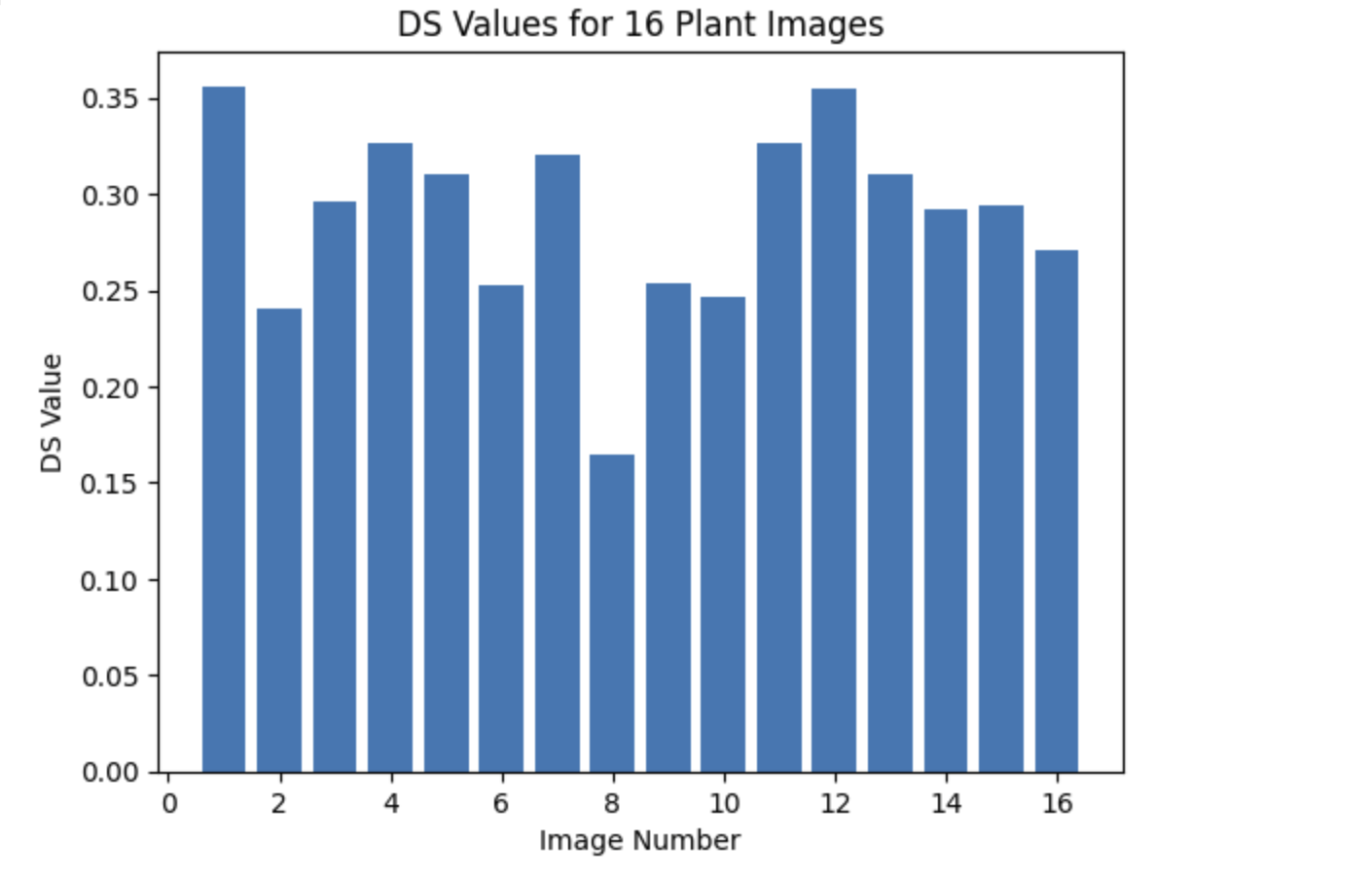
**For all the 16 plant images, please provide a bar graph with x-axis representing the number of the image, and y-axis representing the corresponding DS. 3) Calculate the mean of the DS for all the 16 images.**

* + ulate\_ds function is used to calculate the DS value for each image.
  + The function takes an image as input and processes it to obtain the DS value.
  + The DS value is then appended to the ds\_values list.

1. Plotting the Bar Graph:
   * The plt.bar function is used to create a bar graph.
   * The x-axis represents the image number (ranging from 1 to 16).
   * The y-axis represents the DS values.
   * The range function is used to generate the x-axis values based on the number of DS values.
   * The ds\_values list is passed as the height parameter for the bars.
   * Additional labels and title are set using plt.xlabel, plt.ylabel, and plt.title.
   * Finally, plt.show is called to display the bar graph.

The bar graph provides a visual representation of the DS values for the 16 plant images, allowing for easy comparison and analysis of the green density across the dataset.

**Results :**

****

**Mean DS value: 0.29**

Question 3:

Leaf detection performance evaluation.

1. calculate\_iou:
   * This function takes two bounding boxes as input and calculates the IoU between them.
   * The function computes the intersection area by finding the overlapping region of the two boxes.
   * The union area is obtained by adding the individual areas of the boxes and subtracting the intersection area.
   * The IoU is then calculated as the ratio of intersection area to union area.

Evaluate\_leaf\_detection:

* This function takes two lists of bounding boxes as input: true\_leaves (ground truth) and detected\_leaves (detected by an algorithm).
* It iterates over each true leaf and checks for a matching detected leaf using the IoU threshold (0.5 in this case).
* True positives are incremented when a match is found, and false negatives are incremented when no match is found for a true leaf.
* The false positives are calculated as the difference between the total number of detected leaves and true positives.
* Precision is computed as the ratio of true positives to the sum of true positives and false positives.
* Recall is computed as the ratio of true positives to the sum of true positives and false negatives.
* F1 score is calculated as the harmonic mean of precision and recall

Results :

Precision: 0.50

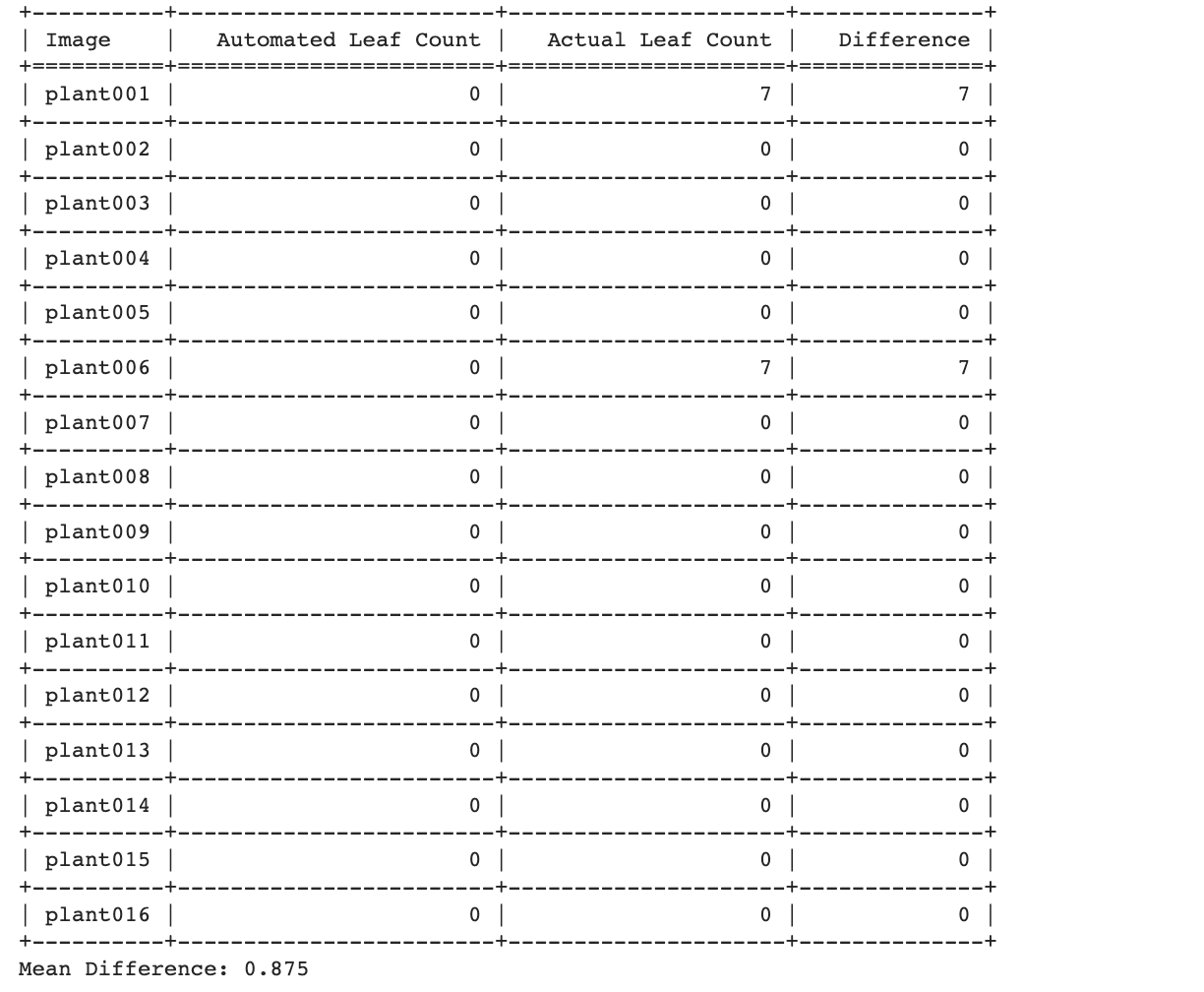
Recall: 0.67

F1 Score: 0.57

**Question 4:**

**Provide a Table showing the absolute difference of the automated leaf counts (multiple detections of the same leaf should count one) and the actual number of leaves provided in the csv file for each image, calculate the mean of the differences in leaf counts over all the 16 images**

1. Load leaf counts from CSV:
   * The code reads the leaf counts from a CSV file using the csv module.
   * The CSV file should contain columns for the plant names and corresponding leaf counts.
2. Create a table:
   * A table is initialized to store the results using the table list.
   * The table structure consists of columns for image name, automated leaf count, actual leaf count, and the difference in counts.
3. Iterate over each image:
   * The code loops through each image in the specified folder.
   * It reads the image using OpenCV's cv2.imread function.
4. Perform leaf counting:
   * The count\_leaves function (not shown in the provided code) is called to perform leaf counting on the input image.
   * The resulting automated leaf count is stored in the automated\_leaf\_count variable.
5. Get actual leaf count:
   * The actual leaf count is obtained from the loaded leaf counts dictionary using the image's plant name.
   * If no entry is found for the plant name, a default value of 0 is used.
6. Calculate absolute difference:
   * The absolute difference between the automated leaf count and the actual leaf count is calculated and stored in the difference variable.
7. Update mean calculation:
   * The difference is added to the total\_difference variable, and the num\_images variable is incremented for mean calculation.
8. Add results to the table:
   * The image name, automated leaf count, actual leaf count, and difference are added as a row to the table list.
9. Calculate mean difference:
   * The mean difference is calculated by dividing the total\_difference by the num\_images.
10. Print the table:
    * The tabulate function from the tabulate module is used to print the table in a formatted grid format.
11. Print the mean difference:
    * The mean difference is printed to the console.

Results : 

Appendix

It is already mention in the code