Assignment 4

Due: April 18

Submission Instructions

- **Create a README file, with simple, clear instructions on how to compile and run your code. *If the TA cannot run your program by following the instructions, you will receive 50% of programing score.***
- **Zip all your files (code, README, written answers, etc.) in a zip file named** $\{firstname\}\_\{lastname\}\_CS5790\_HW3.zip$ **and upload it to Blackboard**

1. (40 points) For this question you will implement the k-means clustering algorithm and apply it to color a given image (*image.png*). you are allowed to use these Python packages: pandas, numpy, skimage.io, matplotlib.

   First use the following code to load the image, which will give you a handle (i.e., img) of a (400, 640, 3) numpy.ndarray. The first two dimensions represent the height and width of the image. The last dimension represents the 3 color channels (RGB) for each pixel of the image.

   ```
   import skimage.io
   import matplotlib.pyplot as plt
   img = skimage.io.imread('image.png')
   skimage.io.imshow(img)
   plt.show()
   ```

   Next implement the k-means algorithm to partition the 400×640 pixels into k clusters based on their RGB values and the Euclidean distance measure. Run your experiment with $k = 2, 4, 7, 10$ with the following given starting centroids:

   $k = 2$: (0, 0, 0), (0.2, 0.2, 0.2)
   $k = 4$: (0, 0, 0), (0.1, 0.1, 0.1), (0.2, 0.2, 0.2), (0.3, 0.3, 0.3)
   $k = 7$: (0, 0, 0), (0.1, 0.1, 0.1), (0.2, 0.2, 0.2), (0.3, 0.3, 0.3), (0.4, 0.4, 0.4), (0.5, 0.5, 0.5), (0.6, 0.6, 0.6)
   $k = 10$: (0, 0, 0), (0.1, 0.1, 0.1), (0.2, 0.2, 0.2), (0.3, 0.3, 0.3), (0.4, 0.4, 0.4), (0.5, 0.5, 0.5), (0.6, 0.6, 0.6), (0.7, 0.7, 0.7), (0.8, 0.8, 0.8), (0.9, 0.9, 0.9)

   For each value of k, you will run k-means until either convergence or your program has conducted 50 iterations over the data, whichever comes first.

Deliverables:

For each $k = 2, 4, 7, 10$, report the final SSE and re-color the pixels in each cluster using the following color scheme:

```
Cluster 1. SpringGreen: (60, 179, 113)
Cluster 2. DeepSkyBlue: (0, 191, 255)
Cluster 3. Yellow: (255, 255, 0)
Cluster 4. Red: (255, 0, 0)
Cluster 5. Black: (0, 0, 0)
Cluster 6. DarkGray: (169, 169, 169)
Cluster 7. DarkOrange: (255, 140, 0)
Cluster 8. Purple: (128, 0, 128)
Cluster 9. Pink: (255, 192, 203)
Cluster 10. White: (255, 255, 255)
```

Submit the SSE values and colored images for each k, together with your program code in your homework submission.

Note: You should normalize the data as a preprocessing step before proceeding with the clustering. Because the range of RGB is [0, 255], so please do the normalization using (R/255, G/255, B/255).

2. (20 points) Consider the following dataset:

```
{ 0, 4, 5, 20, 25, 39, 43, 44 }
```

   (a) Build a dendrogram for this dataset using the **single-link, bottom-up** approach. Show your work.

   (b) Suppose we want the two top-level clusters. List the data points in each cluster.

3. (20 points) Given two clusters

$$C_1 = \{(1,1),(2,2),(3,3)\} \qquad C_2 = \{(5,2),(6,2),(7,2),(8,2),(9,2)\}$$

compute the values in (a) - (f). Use the definition for scattering criteria presented in class. Note that *tr* in the scattering criterion is referring to the trace of the matrix.

   (a) The mean vectors m_1 and m_2

   (b) The total mean vector m

   (c) The scatter matrices S_1 and S_2

   (d) The within-cluster scatter matrix S_W

   (e) The between-cluster scatter matrix S_B

(f) The scatter criterion $\frac{tr(S_B)}{tr(S_W)}$

4. (20 points) A Naive Bayes classifier gives the predicted probability of each data point belonging to the positive class, sorted in a descending order:

Instance #	True Class Label	Predicted Probability of Positive Class
1	N	0.90
2	P	0.82
3	P	0.78
4	P	0.66
5	P	0.60
6	P	0.52
7	N	0.43
8	N	0.42
9	P	0.41
10	P	0.4

Suppose we use 0.5 as the threshold to assign the predicted class label to each data point, i.e., if the predicted probability ≥ 0.5, the data point is assigned to positive class; otherwise, it is assigned to negative class. Calculate the *Confusion Matrix, Accuracy, Precision, Recall, F1 Score* and $Specificity$ of the classifier.