CISC484 Project 3 Report

Peter Chapman

University of Delaware

CISC484: Intro to Machine Learning

Project 3: K-Means Image Segmentation

October 26, 2025

Project Overview

The objective of this project is to create two different versions of image segmentation using four provided images and any personal images. The first task is to implement the image segmentation by directly using the K-means function in scikit-learn. The second task is to implement the K-means method from scratch to perform the same function. For this project, I have essentially split my work up into two parts. One notebook, along with a folder for the created images, has been made and labeled for each of the two tasks.

K-Means Image Segmentation

The first task in this assignment was to use the K-means function in scikit-learn to implement the image segmentation. To create this file, the scikit-learn.org site was consulted in order to learn more about the module and the function. I started this file by importing the relevant libraries and modules in the very first cell. These include the KMeans, imread, imsave, and resize modules. Next, I declared the number of clusters for the data. For this part of the assignment, I chose the number of clusters to be 10, but this could always be changed in future implementations. I then read the image and converted it into a two-dimensional matrix to prepare for implementation. Then, along with the previously identified number of color clusters, the number 10 is chosen for the number of times the algorithm will run with different centroid seeds. After this, the cluster labels and centroid colors are returned for the image. The image is then reconstructed using the colors of the clusters' centroids while also fitting the original image shape. A scaling issue arose before saving the image within the 'remadeImages' folder that I created. For this, I prompted ChatGPT to make sure that the image was scaled correctly. This is identified within the K-means notebook.

Beacon Image Using K-Means

Figure 1 below shows the provided image of the beacon and the reconstructed image using the K-means function. For the most part, the two images look very similar. The reconstructed image, shown on the right, struggled to match the lighting and shadow of the original image. The lighting in the background (around the beacon) changes very abruptly in the segmented version, when it changes more gradually in the original image.

Figure 1: Original image of beacon vs. reconstructed image using K-means function.

Elephant Image Using K-Means

Figure 2 below shows the provided image of the two elephants and the reconstructed image using the K-means function. The reconstructed depiction of the elephants is correct, but the segmented version of the night sky is wrong. In the reconstructed image, it appears as though there are stripes in the night sky, with a mix of blue, orange, and black. In the original image, the changes between these colors are much more gradual, just as in the example with the beacon.

Figure 2: Original image of elephants vs. reconstructed image using K-means function.

Robin Image Using K-Means

Figure 3 below shows the provided image of the robin and the reconstructed image using the K-means function. Just as in the previous two examples, there are issues with the colors in the reconstructed image. The robin's chest is a mix of a few different versions of orange in the segmented version. However, in the original version, the robin's chest is a consistent orange.

Figure 3: Original image of robin vs. reconstructed image using K-means function.

Vulture Image Using K-Means

Figure 4 below shows the provided image of the vulture and the reconstructed image using the K-means function. For the most part, the two images look very similar. The two major problems come with the colors of the bricks and the sky above the vulture. The bricks in the original version are a lively red, but the bricks in the reconstructed version are a very dark color. Additionally, the recreation of the blue sky and the white cloud is not as gradual as it needs to be.

Figure 4: Original image of vulture on the house vs. reconstructed image using K-means function.

Cat Image Using K-Means

The final image I considered for both tasks is an image of my cat Seven sitting in a cabinet drawer. Figure 5 below shows the provided image of my cat and the reconstructed image using the K-means function. The colors for the reconstructed image are wrong in a few different areas. My cat's eyes are supposed to be green, but the recreated image suggests that they are gray. Additionally, there is a mix of gray and white on my cat's chest and paws, when he is all white in these areas.

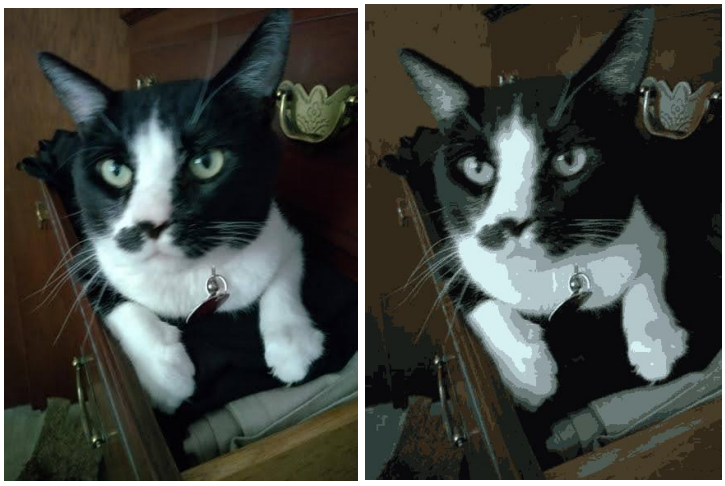

Figure 5: Original image of my cat Seven vs. reconstructed image using K-means function.

Manual Image Segmentation

The second task in this assignment was to manually perform image segmentation without the K-means function. Once again, the scikit-learn.org site was used in order to better understand the K-means function's methodology that I needed to replicate on my own. Additionally, LLM-generated code from ChatGPT was consulted in order to assist the syntax of this operation. I started this file by importing the relevant libraries and modules in the very first cell. These include the imread and imsave modules. Next, I declared the number of clusters for the data. For this part of the assignment, I continued with the choice for the number of clusters to be 10, and this could also be changed in future implementations. To follow this, I created my own version of the K-means function. I set the number of times the algorithm will run to be 20, but it could certainly change for reimplementation. Using the random number generators from the numpy library, clusters were generated randomly for the images. This function then iteratively loops 20 times or until the convergence is reached. Each point is then assigned to the nearest cluster center by using Euclidean distance. New centers are then calculated as the average of the assigned points to that cluster. The mean color of these points is calculated to get the new cluster center. The loop then checks for convergence in order to see if the algorithm has already converged. After all iterations are completed, the function returns the final cluster centers and the labels for the pixels. Just as the K-means notebook, the images are first converted to two-dimensional form. The centers and labels are applied to the image, which are then used to reconstruct the image.

Manual Recreation of Beacon Image

Figure 6 below shows the provided image of the beacon and the reconstructed image using the manual function. The performance results are very similar to the reconstructed image from the K-means function. The reconstructed image, shown on the right, struggled to match the lighting and shadow of the original image. The lighting in the background (around the beacon) changes very abruptly in the segmented version, when it changes more gradually in the original image.

Figure 6: Original image of beacon vs. reconstructed image using manual function.

Manual Recreation of Elephant Image

Figure 7 below shows the provided image of the two elephants and the reconstructed image using the manual function. The performance results are very similar to the reconstructed image from the K-means function. The reconstructed depiction of the elephants is correct, but the segmented version of the night sky is wrong. In the reconstructed image, it once again appears as though there are stripes in the night sky, with a mix of blue, orange, and black. In the original image, the changes between these colors is much more gradual, just as in the example with the beacon.

Figure 7: Original image of elephants vs. reconstructed image using manual function.

Manual Recreation of Robin Image

Figure 8 below shows the provided image of the robin and the reconstructed image using the manual function. The performance results are very similar to the reconstructed image from the K-means function, but possibly even worse. There are major issues with the colors in the reconstructed image for the manual function. The robin's chest is a mix of black and orange in the segmented version. However, in the original version, the robin's chest is a consistent orange.

Figure 8: Original image of robin vs. reconstructed image using manual function.

Manual Recreation of Vulture Image

Figure 9 below shows the provided image of the vulture and the reconstructed image using the manual function. The performance results are very similar to the reconstructed image from the K-means function. For the most part, the two images look very similar. The two major problems come with the colors of the bricks and the sky above the vulture. The bricks in the original version are a lively red, but the bricks in the reconstructed version are a very dark color. Additionally, the recreation of the blue sky and the white cloud is not as gradual as it needs to be.

Figure 9: Original image of vulture on the house vs. reconstructed image using manual function.

Manual Recreation of Cat Image

Figure 10 below shows the provided image of my cat and the reconstructed image using the manual function. The performance results are very similar to the reconstructed image from the K-means function. The colors for the reconstructed image are wrong in a few different areas. My cat's eyes are supposed to be green, but the recreated image suggests that they are gray. Additionally, there is a mix of gray and white on my cat's chest and paws, when he is all white in these areas.

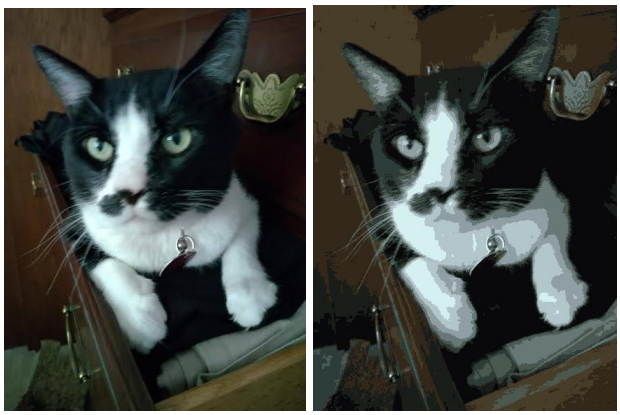

Figure 10: Original image of my cat Seven vs. reconstructed image using manual function.