CS452/552 – Data Science with Python Assignment-3

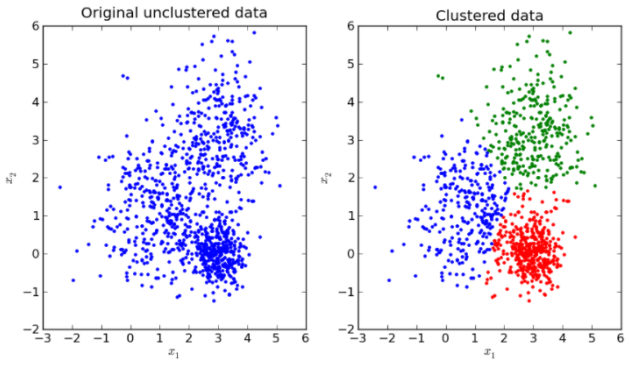
Image Compression by K-Means Clustering

Eyüb Ensari Topcan S018649

Bsc. Industrial Engineering

Introduction

One of the most basic and often used unsupervised machine learning algorithms is K-means clustering.Unsupervised algorithms, on the other hand, make inferences from datasets based solely on input vectors, without than referring to known, or labelled, outcomes.A cluster is a collection of data points that have been grouped together due to particular similarities.You'll set a target number, k, for the number of centroids required in the dataset. A centroid is a fictional or real location that represents the cluster's center.By lowering the in-cluster sum of squares, each data point is assigned to one of the clusters.k-means Clustering Algorithm is a process that minimize the distance between data point from the average data point in the cluster(centroids). We use k-means clustering to extract spam mail and fake news. We also use it to simulate a data point to the closest color tone while rendering. To put it another way, the K-means algorithm finds k centroids and then assigns each data point to the closest cluster while keeping the centroids as small as possible. The average of the data, or determining the centroid, is means in K-means refers.



Source: https://i.stack.imgur.com/clDB3.png

Methodology

We have 5 images as dataset and 3 of the 5 images given to us as a data set were in png and jpeg format. I would convert jpeg format images to png format for convenience when importing, resizing and reshaping. While importing images to my notebook, I created a function that will import the image when the read filename is used as a parameter using the skimage library. and I used this function for *baboon, flowers, lena, umbrella and graffiti* images. Then I converted the size of the images to 256, 256 format using the resize method of the OpenCV library. So they all have the same format and it will be easier to process. Since the images I imported are kept in variables in the form of an array according to their r, g, b values, I did the following operations with these variables. For example, I made the operations easier by converting the type of these arrays to integer.

Then I created a reshape function by taking the shape values of rows and columns of all images. This function made the number of rows of each image equal to the number of resolution of the image. so I got it as rows*columns. So we have a 2D array. Columns values are 3 for rgb values. And I applied it to all images. However, I noticed that the flowers.png image has 4 columns and I created an empty array of (256, 256, 3) and created this array with the flower array. I then used the getsize() method to measure how many bytes each image uses in computer memory. byte is the measure of the array as 0-1 and I have printed how much space our images have use. My next step was to find the different colors, so I created the unique color function and printed the unique array values in each image.

In order to apply K-means clustering, I chose 8 different k values, these are 256, 128, 64, 32, 16, 8, 4, 2. I applied these values one by one for Baboon.png. First of all, I ran the Kmeans method with the value of k=256 and fit the arrays I reshaped into the model I created. Then I kept the cluster center of the model I created in the variable I created with the compressed tag and visualized it by reshaping. The process I did here was to create 256 separate cluster centers and ensure that the center gets its color at the closest points to that center.

Then, I calculated the amount of memory used by the baboon image after the 256 kmeans model, in bytes, and kept it in the baboon_bytes list. I also applied these operations on k values of 128, 64, 32, 16, 8, 4 and 2 values. I applied these 8 different k values to flowers, lena, umbrella and graffiti images. As the k value decreased, the number of clustered clusters decreased, so the colors decreased and the shapes became incomprehensible.

Implementation Details

When we use the terms classification and clustering, which are machine learning methods, there are some metrics to evaluate our model. Thanks to these metrics, we can measure the accuracy and consistency of our model by learning how accurate it works and the scores of the results. These results allow us to make better decisions.

WCSS: Within Cluster Sum of Squares is the mean of the square of the distances between the data points and the Cluster Centroids. In order to calculate it, we subtract the coordinate of the image's data points from the centroid point, take its square, and divide by the image's data points and get the average.

- **Within Cluster Sums of Squares :** $$\mathrm{WSS} = \sum_{i=1}^{N_C} \sum_{x \in C_i} d(\mathbf{x}, \bar{\mathbf{x}}_{\mathbf{C_i}})^2$$

C_i = Cluster, N_c = # clusters, $\overline{x}_{c_i}$ = Cluster centroid,

BCSS: It is found by multiplying the square of the distance between the specified centroid point and the sample mean value by the number of data points.

- **Between Cluster Sums of Squares:** $$\mathrm{BSS} = \sum_{i=1}^{N_C} |C_i| \cdot d(\bar{\mathbf{x}}_{\mathbf{C_i}}, \bar{\mathbf{x}})^2$$

C_i = Cluster, N_c = # clusters, $\bar{x}_{c_i}$ = Cluster centroid, $\bar{x}$ = Sample Mean

I kept the bytes, wcss and bcss values you calculated in separate lists for each image. Then I applied the Elbow Method by sorting the wcss and bcss scores in ascending order of their k values

Explained Variance

$$explained\ variance(y, \hat{y}) = 1 - \frac{Var(y - \hat{y})}{Var(y)}$$

Slhouette Coefficient

- **a**: The mean distance between a sample and all other points in the same class.
- **b**: The mean distance between a sample and all other points in the *next nearest*

The Silhouette Coefficient *s* for a single sample is then given as:

$$s = \frac{b - a}{max(a, b)}$$

Then I printed the color name from the rgb color code of the cluster center for each k value. At this stage, I used the CSS3 color codes of the webcolors library.

Before we talk about how much space the images use, we need to know that these values are carried in arrays as numeric and are encoded according to the rgb code. In the snippet below, it is given how many bytes the images are in their original state.

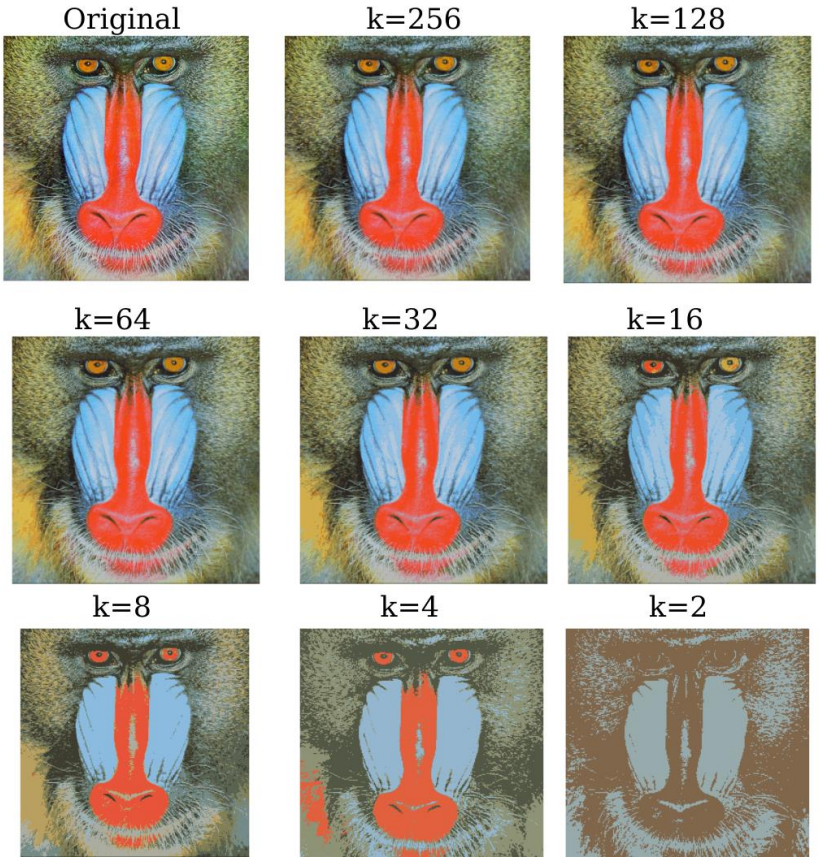
baboon bytes 651142 baboon unique colors 62070

flowers bytes 615128 flowers unique colors 57848

graffiti bytes 6254727 graffiti unique colors 45614

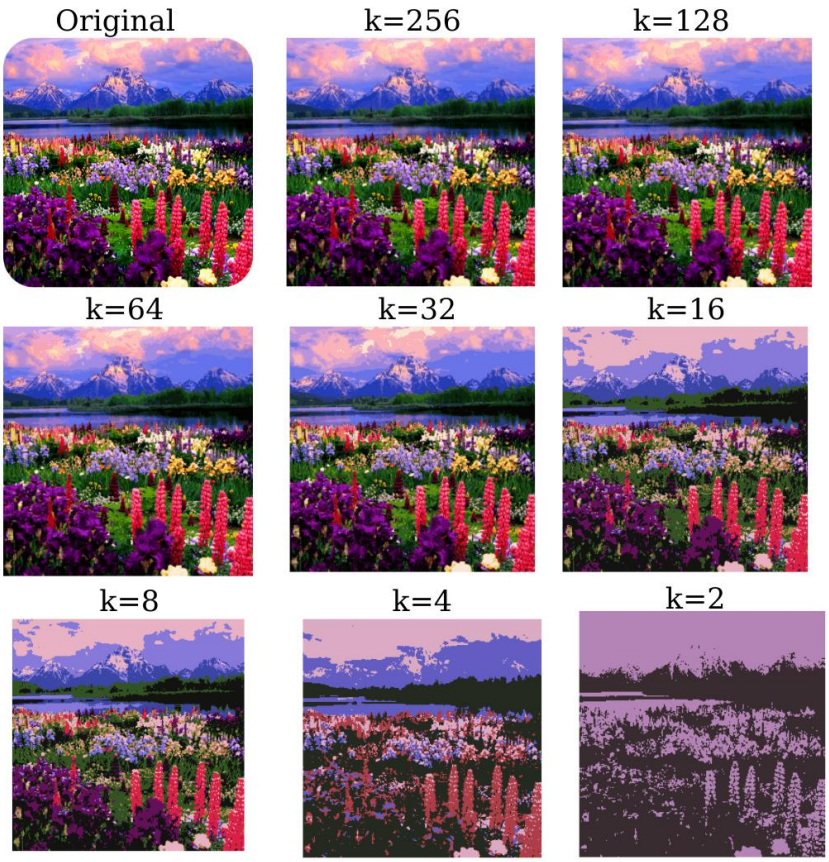
lena bytes 473831 lena unique colors 48331

umbrella bytes 1608086 umbrella unique colors 48926

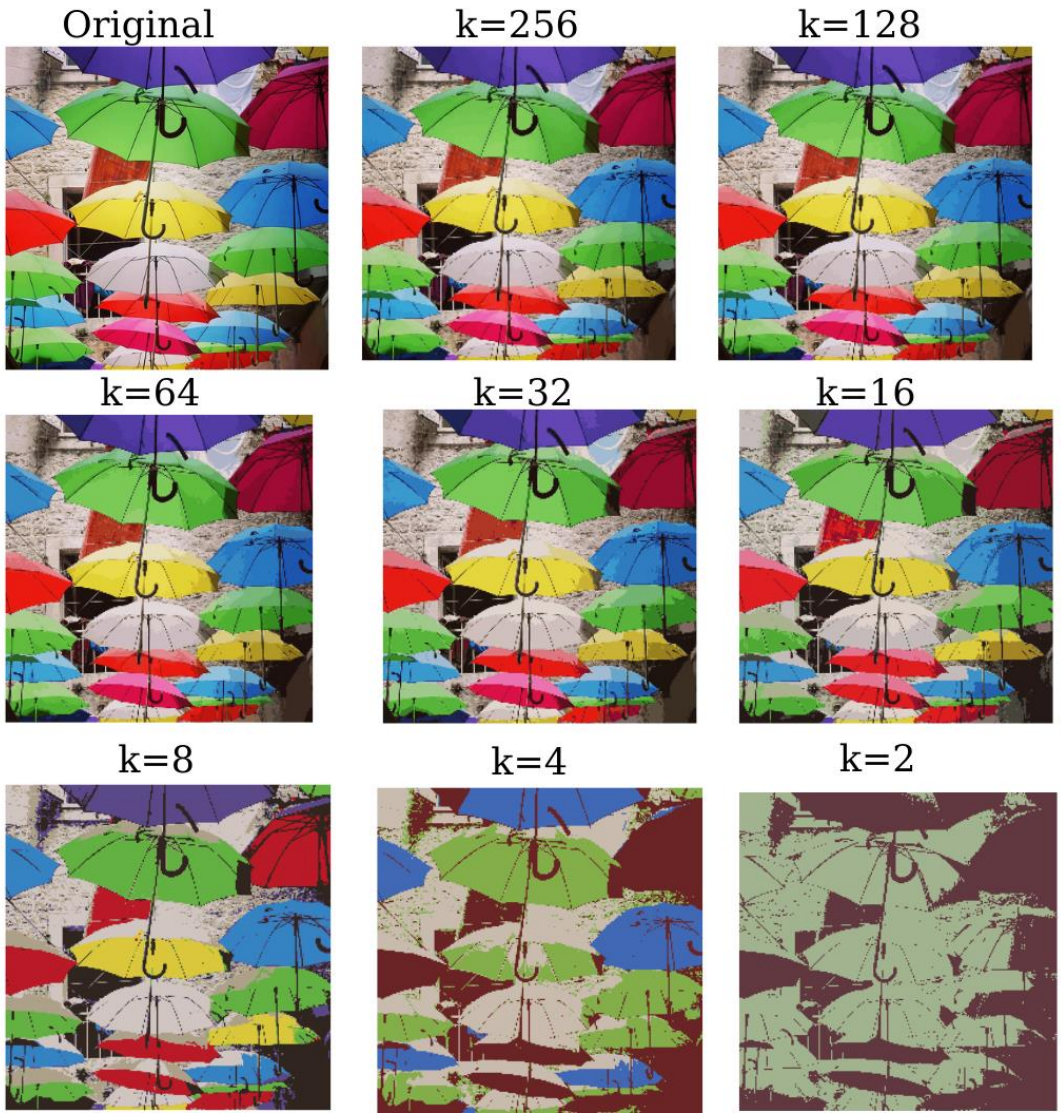


We see 9 different versions of the baboon image. First of all, the original baboon image draws attention with its high bytes and high quality. Then, when we apply kmeans clustering with k=256, the quality of the image decreases, even if it is not visible, and when the k=128 value is applied, the quality of the picture decreases. Then, when the kmeans clustering algorithm is applied with the value of k=64, we can recognize the image and its byte becomes much lower. At k=32, it is noticed that the quality of the picture has deteriorated considerably, and at k=16, the picture has become of poor quality. The first k value is k=8, where the colors blend into each other and the picture becomes meaningless, because the red areas become a single color, the 3D perception in the picture is lost and the picture takes up much less space. At k=4, the picture becomes pale and the unique color values are reduced, being limited to 4 pieces. Finally, when the k=2 value is applied, only 2 colors are used and the picture becomes unrecognizable.

Kmeans clustering, as we mentioned in the introduction, restricts the data points to a certain number and clusters the closest other data points to the centroids. We witnessed this process by reducing the number of colors in the baboon image. Let us remind you that the number of colors is equal to the value of k.

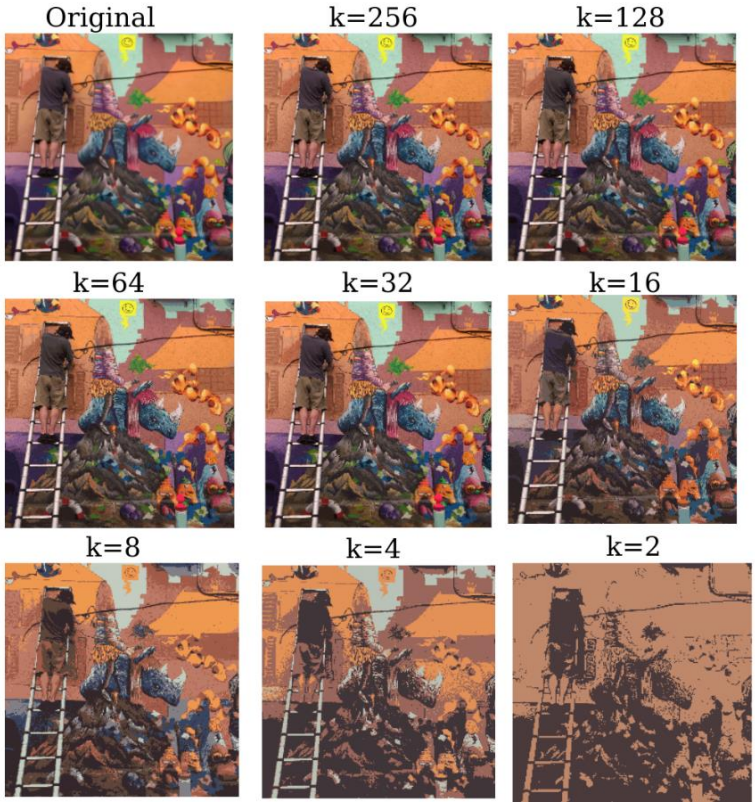


In the flowers image, where it is the same as in the baboon image, the intelligibility of the picture is not lost until k = 8, but at k=4 and k=2 values, the picture becomes unrecognizable and loses its original quality.



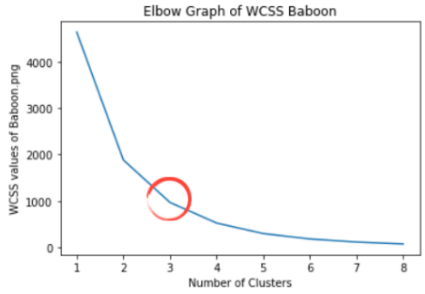
As with the other two reisms, umbrellas image loses its 3D image up to k=8. At k=4, it is noteworthy that while red and yellow colors disappear, green and blue umbrellas can be displayed. With a value of k=2, it means that the most used color is a shade of green.

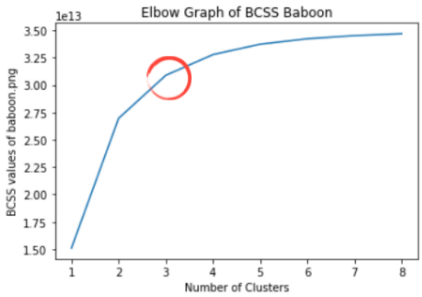

Original
k=256
k=128
k=64
k=32
k=16
k=8
k=4
k=2

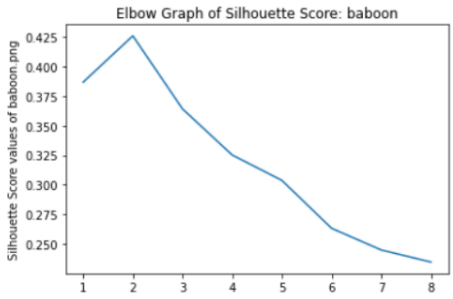

Original
k=256
k=128
k=64
k=32
k=16
k=8
k=4
k=2

Results

The Elbow method is a method that enables the determination of breakpoints when we plot the wcss values. This is the insight of decreasing image quality and decreasing byte values with decreasing k value in this graph. In order to choose the most optimal k value, we need to estimate the breakpoint correctly. The BCSS graph can also be viewed at very close k values. As the Silhouette Coefficient increases, the optimal value increases and the picture with the highest score is selected.

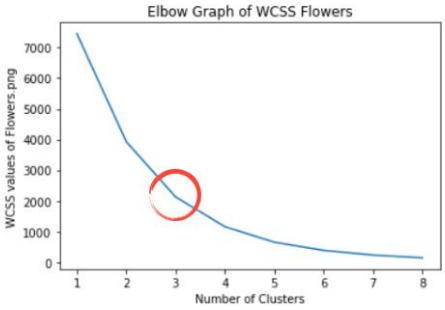


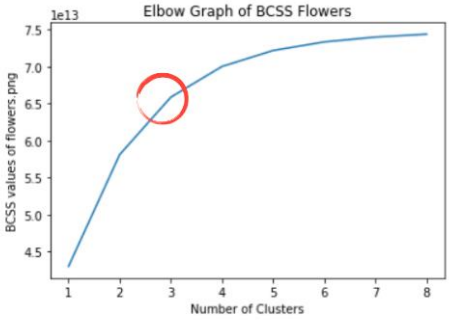


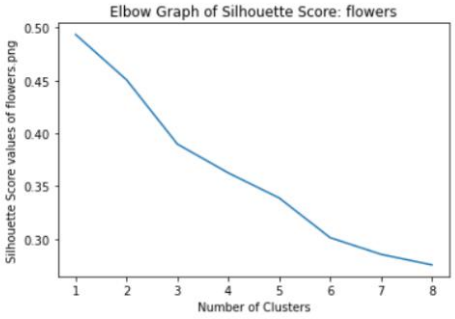


When we look at baboon's WCSS graph, it shows a break at the 2nd k value, but since its slope is lower than the 3rd break point, the 3rd break point, that is, the k=64 value, is an optimal image with both the lowest byte and the highest quality.

	k Value	Image Bytes	WCSS	BCSS	Silhouette Coefficient
0	256	139251	4633.811167	1.509674e+13	0.386851
1	128	122197	1881.748745	2.696503e+13	0.426047
2	64	97407	966.084274	3.089790e+13	0.364207
3	32	68332	522.732325	3.279109e+13	0.325187
4	16	44662	295.892282	3.374279e+13	0.303719
5	8	27878	179.711279	3.423942e+13	0.263192
6	4	15395	113.227698	3.452812e+13	0.244817
7	2	9427	72.026366	3.470138e+13	0.234600

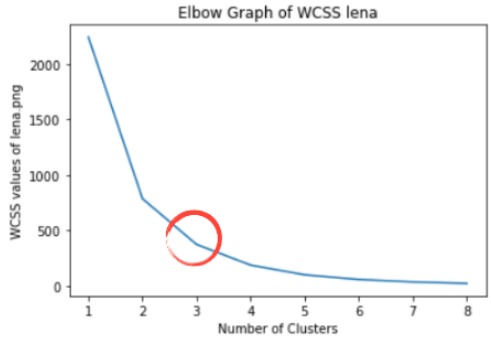


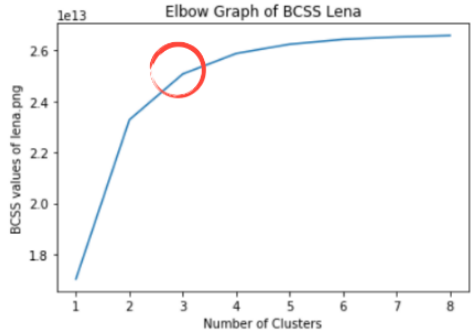


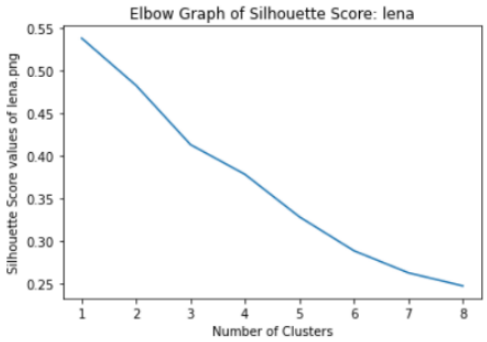


When we look at the WCSS graph of Flowers, it shows a break at the 2nd k value, but since its slope is lower than the 3rd break point, the 3rd break point, that is, the k=64 value, is an optimal image with both the lowest byte and the highest quality. Still, Silhouette looks at the score, and the decision must be made.

	k Value	Image Bytes	WCSS	BCSS	Silhouette Coefficient
0	256	123173	141.870096	7.439726e+13	0.275469
1	128	108147	233.311151	7.400339e+13	0.285409
2	64	87154	384.762679	7.334299e+13	0.301255
3	32	65160	651.152316	7.220837e+13	0.338819
4	16	43400	1154.636862	7.005187e+13	0.362556
5	8	26032	2128.081096	6.584429e+13	0.389836
6	4	14116	3930.105316	5.815378e+13	0.450765
7	2	7558	7451.615352	4.299531e+13	0.493659

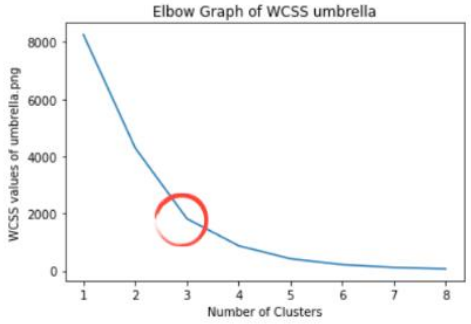


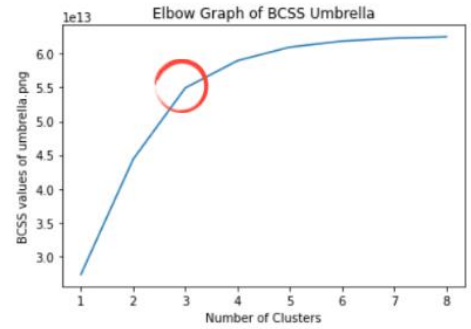


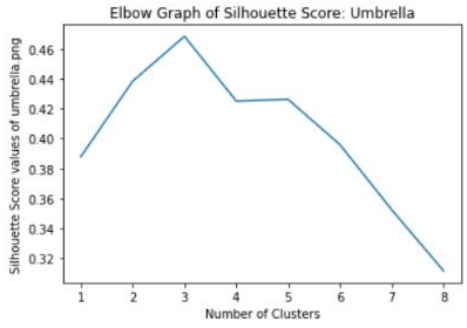


If we look at Lena Image's WCSS graph, it shows a break at the 2nd k value, but since its slope is lower than the 3rd break point, the 3rd break point, that is, the k=64 value, is an optimal image with both the lowest byte and the highest quality. Still, Silhouette looks at the score, and the decision must be made. And according to the Silhouette Coefficient, the 3rd k value, that is, k=64, is the most optimal.

	k Value	Image Bytes	WCSS	BCSS	Silhouette Coefficient
0	256	98011	22.851251	2.658228e+13	0.246926
1	128	82144	36.258789	2.652131e+13	0.262272
2	64	63082	58.703406	2.642488e+13	0.288225
3	32	44147	100.555945	2.624453e+13	0.327985
4	16	28663	187.793837	2.587238e+13	0.377891
5	8	18788	374.232656	2.508385e+13	0.412792
6	4	10585	786.313326	2.332820e+13	0.482355
7	2	4706	2242.249686	1.705161e+13	0.537884

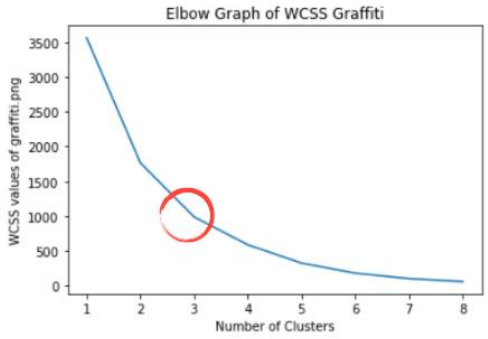


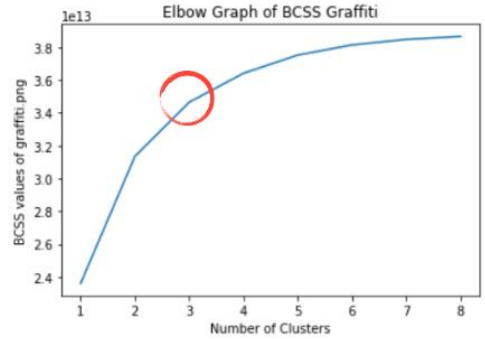


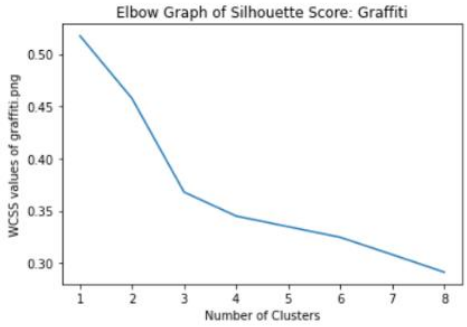


When we look at the WCSS chart of Umbrella Image, it shows a break at the 2nd k value, but since its slope is lower than the 3rd break point, the 3rd break point, that is, the k=64 value, is an optimal image with both the lowest byte and the highest quality. Still, Silhouette looks at the score, and the decision must be made. And according to the Silhouette Coefficient, the 3rd k value, that is, k=64, is the most optimal.

	k Value	Image Bytes	WCSS	BCSS	Silhouette Coefficient
0	256	83962	69.774271	6.250168e+13	0.311748
1	128	69733	118.804865	6.229720e+13	0.352484
2	64	51491	215.330608	6.185278e+13	0.395985
3	32	35859	426.139658	6.096688e+13	0.426227
4	16	26995	876.402117	5.901232e+13	0.425109
5	8	17416	1824.074470	5.495328e+13	0.468306
6	4	11125	4296.305053	4.436502e+13	0.438331
7	2	6522	8256.576879	2.732952e+13	0.387933







Likewise, when we look at the WCSS graph of Graffiti Image, it showed a break at the 2nd k value, but since its slope is lower than the 3rd break point, the 3rd break point, that is, the k=64 value, is an optimal image with both the lowest byte and the highest quality. is. Still, Silhouette looks at the score, and the decision must be made. And according to the Silhouette Coefficient, the 3rd k value, that is, k=64, is the most optimal.

	k Value	Image Bytes	WCSS	BCSS	Silhouette Coefficient
0	256	115419	60.915050	3.865623e+13	0.291287
1	128	96728	102.929404	3.847275e+13	0.308088
2	64	78329	180.672638	3.814271e+13	0.324665
3	32	61957	328.921896	3.750615e+13	0.334804
4	16	43509	587.097209	3.636762e+13	0.344991
5	8	28070	988.796155	3.465736e+13	0.367874
6	4	14870	1764.546744	3.133101e+13	0.457504
7	2	7897	3561.984802	2.362165e+13	0.517483

Conclusion

In our research, we converted 5 different images in 3 png and 2 jpeg formats to png format and imported them into our notebook. Then, we brought our images to workable and suitable dimensions with the reshape and resize methods. Then we provided unique color and byte control. The resolution values are kept in the rows of the image_arrays we obtained, and the red, green, blue, i.e. rgb values in the columns. When these values are combined, the color of each pixel forms the whole image. The Kmeans algorithm, on the other hand, creates clusters as many as the k value given and equates the color of the centers (centroids) of these clusters to the sample of the closest data points, and the number of colors used by the image decreases. Since the data that the image array carries decreases, the area it uses, namely the byte, is decreasing. Thus, the quality of the image, which takes up less space, also decreases. In this research, we have optimized both low-byte and high-quality images by using metrics. Using this method, the quality of the pictures is reduced and cloud services are created by increasing them after sending them. We used the elbow method to find the optimal value in the metrics we used. We applied the Elbow method to the wcss graph and decided whether it was optimal by looking at the silhouette value of the points with diffraction.

Refferenes

https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1

https://www.kaggle.com/achintyatripathi/kmeans-from-scratch-with-silhoutte-and-elbow-curve

https://towardsdatascience.com/image-compression-using-k-means-clustering-aa0c91bb0eeb

https://auth0.com/blog/image-processing-in-python-with-pillow/

https://www.w3resource.com/numpy/manipulation/resize.php

https://www.geeksforgeeks.org/python-numpy-numpy-resize/

https://medium.com/codex/rgb-to-color-names-in-python-the-robust-way-ec4a9d97a01f

https://stackoverflow.com/questions/68714612/compute-between-clusters-sum-of-squares-bcss-and-total-sum-of-squares-manually

https://www.analyticsvidhya.com/blog/2021/08/kmeans-clustering/

https://stackoverflow.com/questions/46540831/how-to-read-an-image-in-python-opencv

https://en.wikipedia.org/wiki/Elbow_method_(clustering)

https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/

https://www.scikit-yb.org/en/latest/api/cluster/elbow.html#:~:text=K%2Dmeans%20is%20a%20simple,number%20(k)%20of%20clusters.&text=The%20elbow%20method%20runs%20k,average%20score%20for%20all%20clusters.