FINAL PROJECT

K-Means Clustering Analysis and its Applications

– DONTHULA SRINISH (EE20BTECH11015)

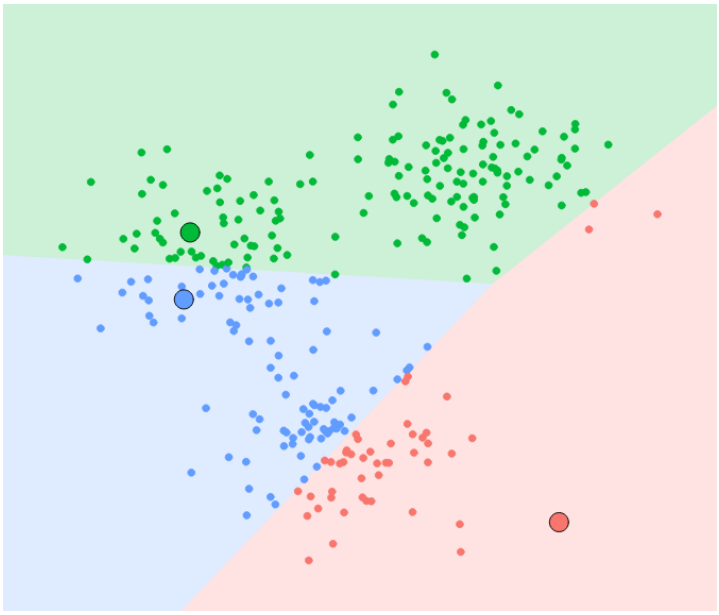
– NAROPANTH SRIKAR RAO (EE20BTECH11035)

INTRODUCTION:

K-means clustering is a popular unsupervised machine learning algorithm used to group data points into k clusters based on their similarity. Basically, it finds groups which are similar in an unlabelled data. K-means is commonly used in fields such as image recognition, image compression, market segmentation, and anomaly detection. However, it requires careful consideration of the number of clusters to use and can be sensitive to initial centroid selection.

One important consideration in K-means clustering is choosing the optimal number of clusters (k) for the data. There are several methods to determine k, such as the **Elbow method**, **Silhouette method**, or gap statistic. One limitation of K-means clustering is that it assumes spherical clusters with equal variance. If the data has non-spherical clusters or varying cluster densities, other algorithms such as DBSCAN or Gaussian mixture models may be more suitable.

In this following picture, we can clearly differentiate the data into 3 different different clusters. Each cluster can be identified through a different color. Each feature can be categorized based on the region it falls under one of these clusters. Also, the big circles in the picture are the cluster centers. These cluster centers are relocated such that all the data points in the nearby region are closer to this cluster center compared to other cluster centers.



ALGORITHM :

The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. We will randomly initialize the K cluster centroids as K random points in the dataset.
The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

This algorithm is already implemented in the scikit-learn package i.e we can apply K-means by importing sklearn.cluster.KMeans.

ELBOW METHOD:

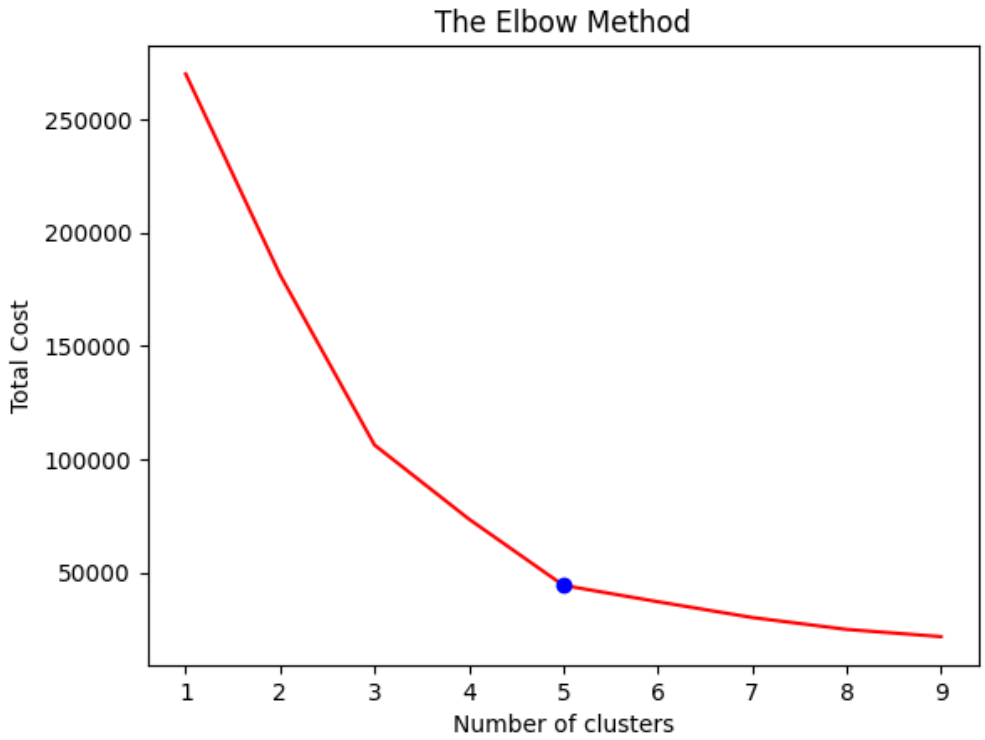
As discussed earlier, the most important thing is to choose the optimal number of clusters(K) for a given data. There are several methods to choose the optimal value for K and the Elbow method is one of the most widely used methods to find it.
K-Means algorithm finds the clusters by minimizing the mean square distance between cluster centroids and the data points. In the Elbow method, we run the K-Means algorithm for different values of K and plot cost(inertia) function vs K. From the plot, we get an elbow shaped curve and the optimal value for K is chosen at the elbow or the maximum slope varying point.

ANALYSIS OF K-MEANS CLUSTERING:

Firstly, we considered a dataset of different people with their annual income and spending score(1-100) . This is an unlabeled data. The people can be grouped as:

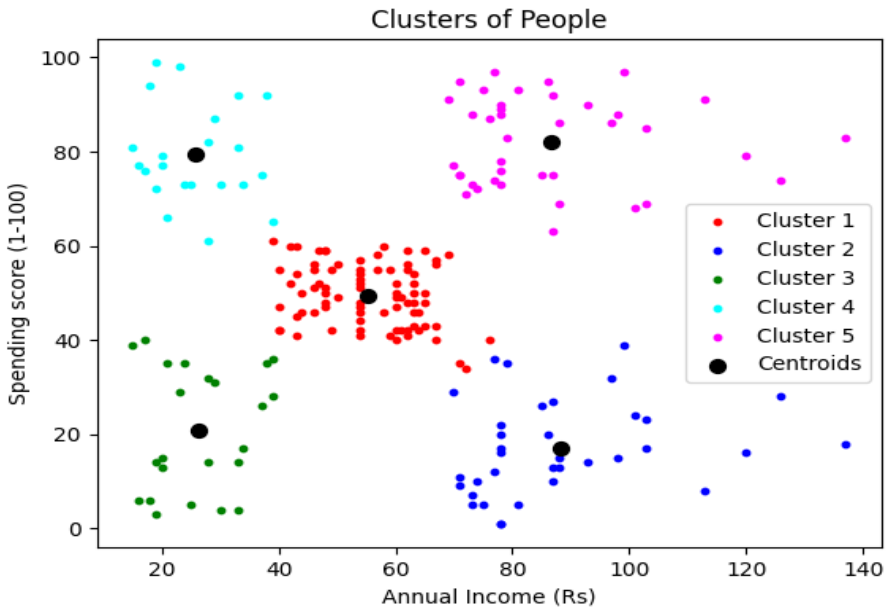
- Annual income is high and spending score is high
- Annual income is low and spending score is high
- Annual income is high and spending score is low
- Annual income is low and spending score is low

The rest of the people can be categorized into a separate category where both annual income and spending score is close to average.
To find the number of groups among the people according to their annual income and spending score first we used the elbow method. By implementing the K-Means algorithm on this dataset using scikit-learn and plotting cost(inertia) function for different values of K we got:



From the above plot, we can see that the curve changes its slope abruptly at 2 different points. But, in order to minimize the total cost, we consider the number of clusters(K)=5. By considering K=5, we now implement the K-Means clustering algorithm and plot different clusters.

In this plot, different clusters are marked with different colors and all the five cluster centroids are marked with black color. The red cluster would have the most average spending score and average annual income.



SILHOUETTE METHOD:

The Silhouette method is a clustering evaluation technique used to measure the quality of a clustering solution. It provides a way to assess how well each data point fits into its assigned cluster by computing a silhouette score.

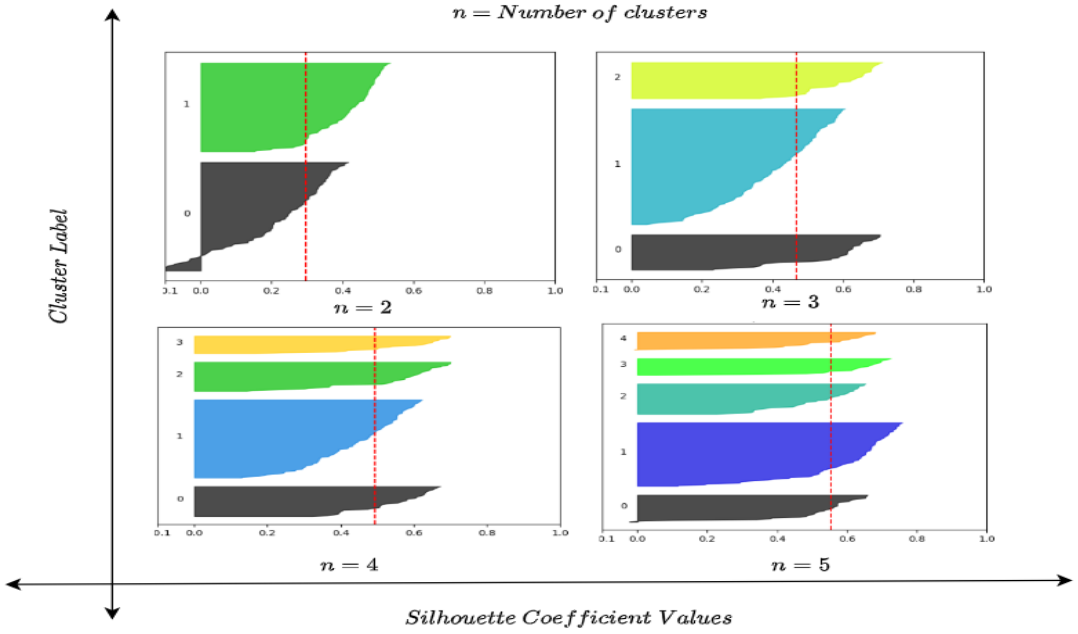
The silhouette score for a data point measures how similar it is to its own cluster compared to other clusters. It is calculated as the difference between the average distance of a data point to all other points in its own cluster (a) and the average distance to all points in the nearest neighboring cluster (b), divided by the maximum of the two distances:

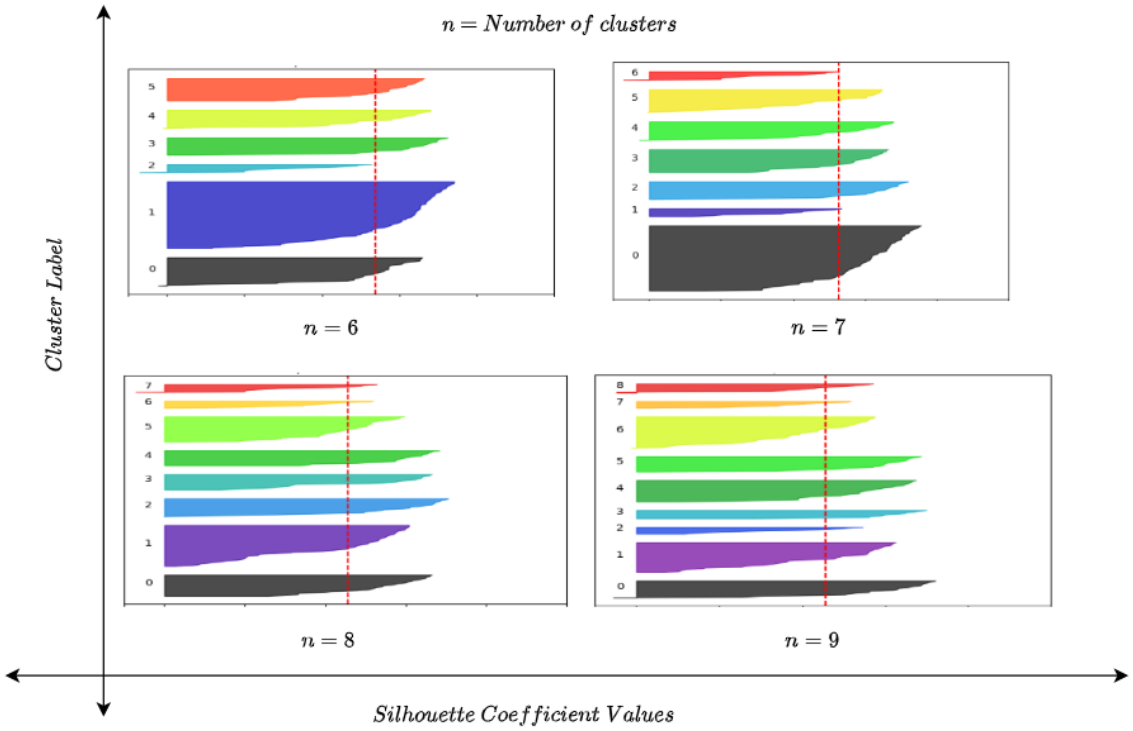
silhouette score = (b - a) / max(a, b)

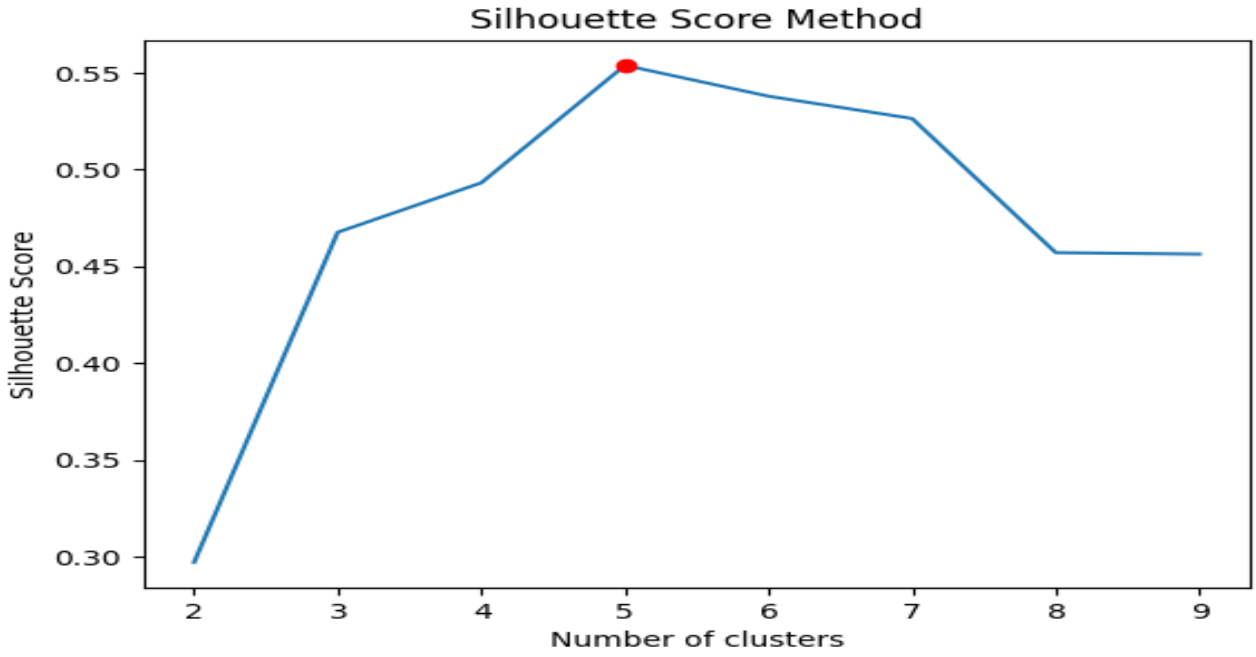
The silhouette score ranges from -1 to 1, where a score of 1 indicates a well-clustered data point that is far from other clusters, and a score of -1 indicates a data point that has been assigned to the wrong cluster.To apply the Silhouette method, we compute the average silhouette score for all data points in a clustering solution. Higher average silhouette scores indicate a better clustering solution with well-separated clusters and appropriate data point assignments.

The Silhouette method is often used in conjunction with other clustering evaluation techniques, such as the elbow method or the gap statistic, to select the optimal number of clusters for a given dataset.

Now, let us implement this method on our dataset. These are the following results:







Clearly, from the silhouette score vs number of clusters plot, the dataset has achieved its maximum silhouette score at K = 5. Even through the elbow method we got the number of optimal clusters as 5. This shows that both are optimal methods.

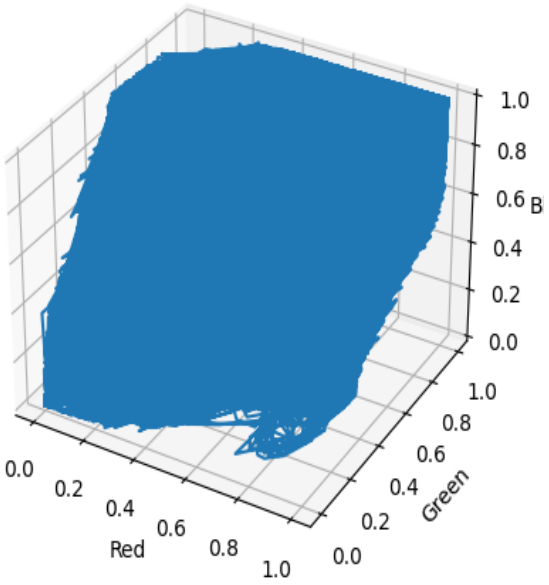
Now, let us apply this K-Means algorithm on **Image Compression**. Let us see the change in colors after applying K-Means for different values of K.

IMAGE COMPRESSION:

We can use K-Means Algorithm to compress an Image. An Image consists of pixels where each pixel has some color. Each pixel is made up of a red, green and blue subpixel that lights up at different intensities to create different colors. The specific color information that a pixel describes is some blend of three components of the color spectrum -- RGB.

In a straightforward 24-bit color representation of an image, each pixel is represented as three 8-bit unsigned integers (ranging from 0 to 255) that specify the red, green and blue intensity values. This encoding is often referred to as the RGB encoding.

We have taken a bird image to compress it. We have scaled the red, green and blue intensity values of pixels to 1 by dividing 255.



The total number of pixels in the above image are 950*950 which is 9,02,500 pixels. Each pixel has 3 intensity values of red, green and blue. The intensity values(scaled to 1) of the pixels are plotted above.

Now we have a data of 9,02,500 pixels which are plotted in 3D. We can group the data points(intensities of pixels) into K-clusters (K colors) in such a way that the intensities of red, green and blue are similar or nearer. So, to compress the image, we can replace the group of pixels with a pixel that has an intensity value which is the average of the intensities of the group. Instead of a group of pixels, we are making it to one pixel which is compression of the image.

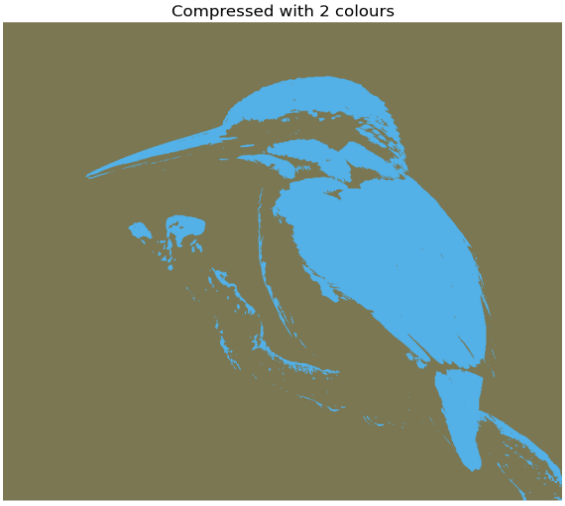
To do this, we have used the K-Means Algorithm which is basically grouping the data points. Here we are grouping colors with different intensities. To implement this algorithm, we can use the scikit-learn package. But, we have implemented the algorithm from scratch using numpy and matplotlib for better understanding of the algorithm.

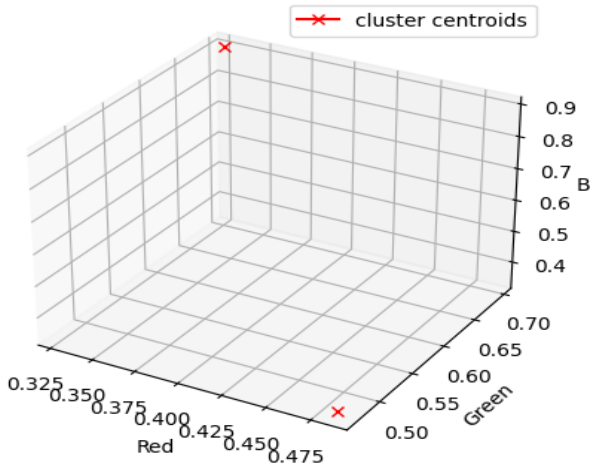
We have divided the algorithm into some parts by implementing different functions and then finally using all of them we can run the algorithm. The functions that we implemented are:

```
KMeans_init_centroids(X, K)--> At the start of the K-Means algorithm, we
need to initialize the cluster centroids. This function randomly
initializes K cluster centroids by taking K data points from X. It returns
the K cluster centroids.

find_closest_centroids(X, centroids)--> This function will find the
closest centroid for each of the data points by computing the distance
from all centroids and then consider the centroid with minimum distance as
the closest centroid. It returns the index of the closest centroid for
each data point in X.

compute_centroids(X, index, K)--> After finding the closest cluster
centroid for each and every point in X, we need to update the cluster
centroids as the mean of the data points having same index. It returns the
updated cluster centroids after each iteration of the KMeans Algorithm.

run_KMeans(X, initial_centroids, max_iters)--> The KMeans Algorithm is
carried out in this function. In every iteration, we find the closest
clustered centroids for all data points and the updated cluster centroids.
After the max_iters, we are done with the algorithm. The pixels have been
grouped according to the value of K(K colors). It return the compressed
pixel values of the image.
```

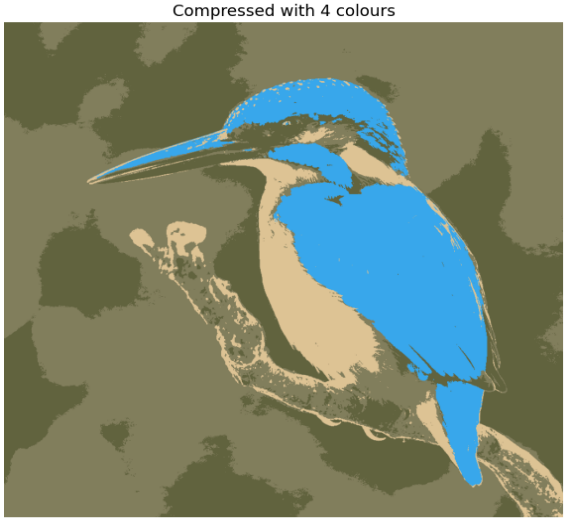
RESULTS:

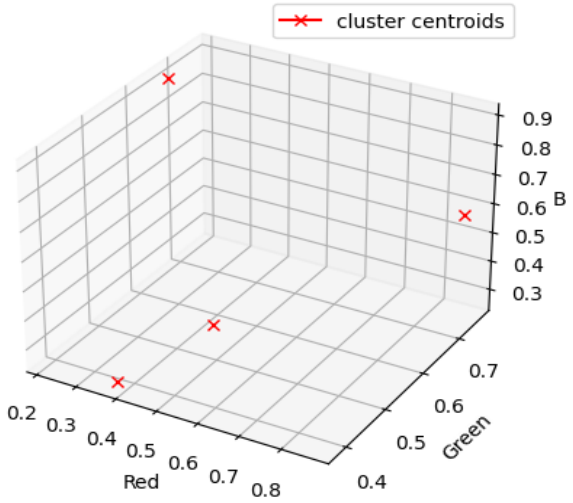
Compressed with 2 colours



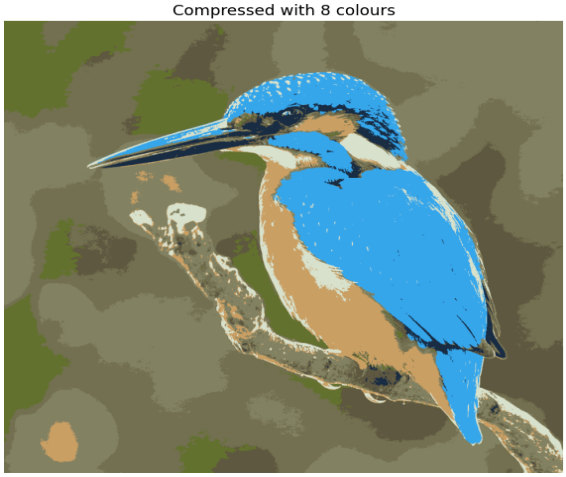


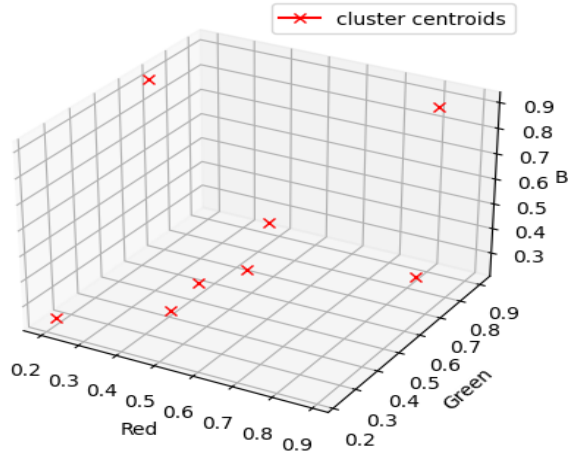
Compressed with 4 colours

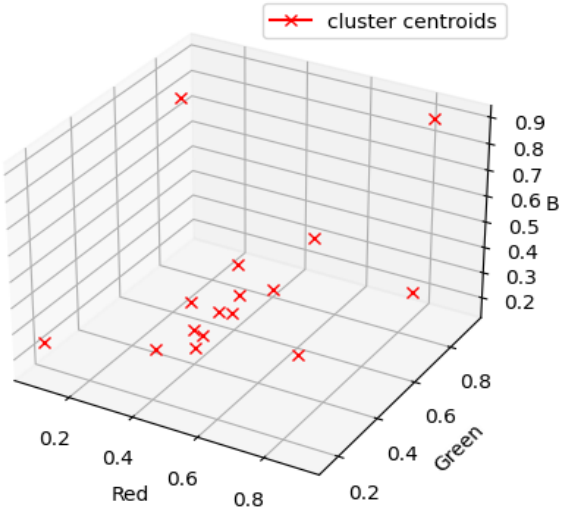


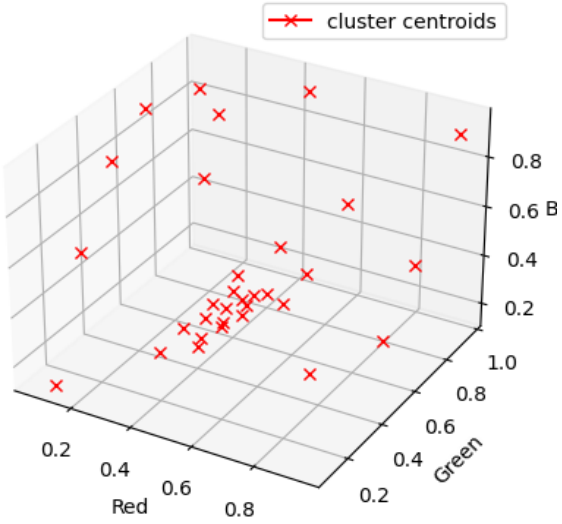


Compressed with 8 colours









After implementing the above functions, we can understand the KMeans algorithm in a broad way. After implementing the algorithm for the above image for different values of K(colors) 2, 4, 8, 16 and 32, we got the above compressed images.

From the above images, we can see that the image is compressed for different values of K which means different numbers of colors. When K=2, we can clearly see that the image is compressed to only 2 colors. As the value of K increases, the number of colors increases which means more clusters. We can see that clearly in the above images.

CONCLUSION:

- To find the optimal number of clusters we can use both elbow method and silhouette method.
- In summary, k-means clustering is a powerful and versatile algorithm that can be used for a variety of applications. However, it is important to carefully consider the choice of K and to be aware of the assumptions and limitations of the algorithm.
- K-Means clustering is a powerful technique for compressing images, but the quality of the resulting compressed image can be highly dependent on the number of centroids used and the choice of distance metric. Careful experimentation and tuning are required to obtain the best results.

GITHUB LINK FOR CODES:

- https://github.com/Srinish27/Data_Science_Analysis_Project

REFERENCES:

- https://en.wikipedia.org/wiki/Silhouette_(clustering)
- https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning
- https://drive.google.com/file/d/1nkQMQzbMMlGXX4h6i9eS8MjDFrUzcVlJ/view?usp=sharing
- https://en.wikipedia.org/wiki/Elbow_method_(clustering)#:~:text=In%20cluster%20analysis%2C%20the%20elbow,number%20of%20clusters%20to%20use.