Clustering

Unsupervised Learning Final Project
Sep, 2023

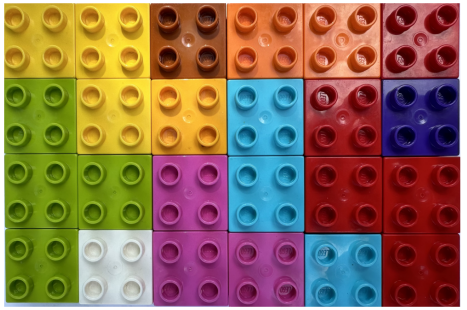
1. Objective

Image Segmentation:

Image segmentation is very useful for image recognition because it extracts objects of interest for further analysis. It is an important step in more advanced machine learning models like deep learning.

Goal:

To implement 3 different clustering algorithms to study image segmentation.



Data:

We will be analyzing a picture of LEGO blocks placed on beside the other.

2. LEGO Image

Dataset Description

This image has the following attributes:

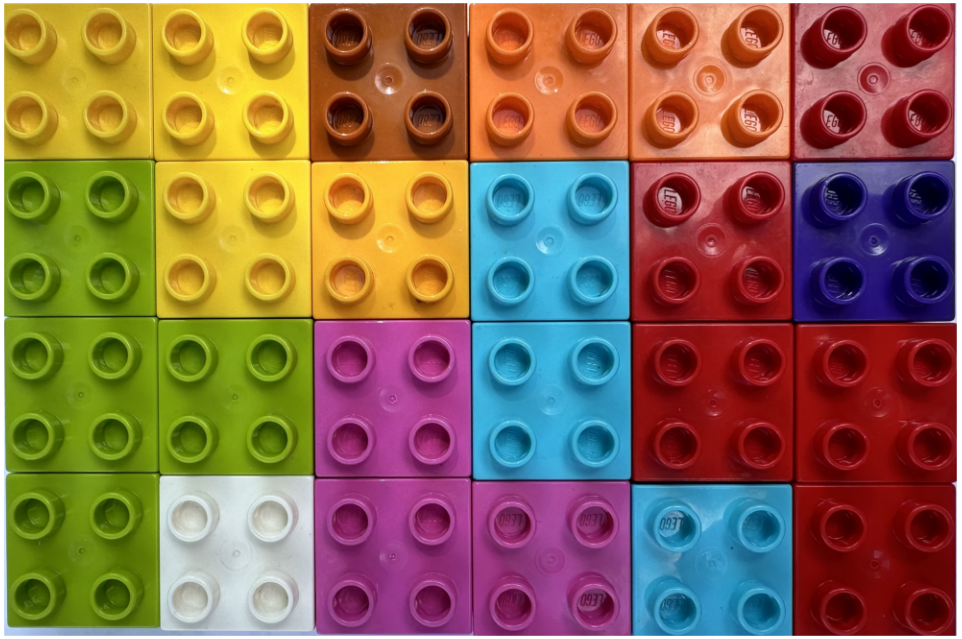


Image Size:

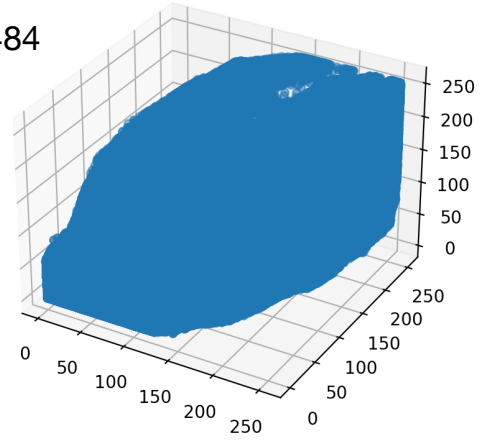
2178 x 3278 pixels with (R,G,B) values for each of these pixels.

Flattened Image Size:

rows = 2178 x 3278 = 7139484

Column = R,G,B values = 3

3D distribution of pixels :



3. Data Pre-Processing

The shape of the original image is in form of **3d** array of shape [n_p , n_p , 3].

Where, (n_p x n_p) is a 2x2 grid of number of pixels and 3 represents the RGB values for each pixel.

Before we pass on the this image into our clustering algorithms, we need to flatten the image and convert it into an 2-dimensional array with all the pixels as rows and the RBG color values as columns.

The shape of the flattened image will, therefore, be an **2D** array of shape [(n_p xn_p) , 3]

4. Clustering Methods Implemented

We have implemented the following clustering algorithms to segment (breakdown) the original image into several clusters (groups) :

KMeans clustering

Gaussian Mixture

Mean Shift

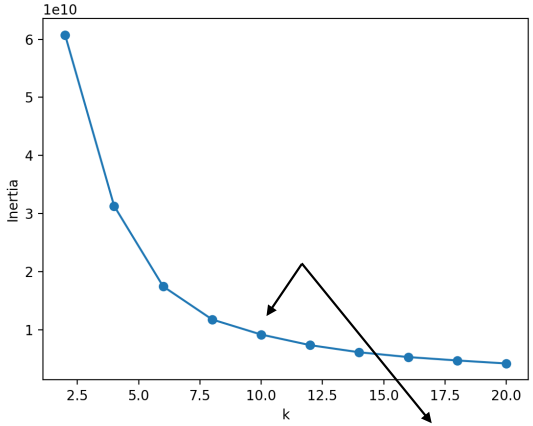
4.1 KMeans

1. The KMeans algorithm was run for cluster values, k, ranging from 2 to 20.

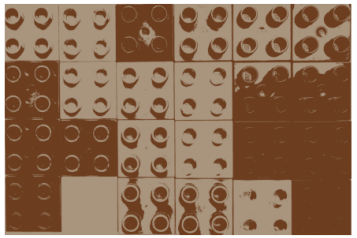
2. The elbow plot, on the right, indicates that the optimal value for the number of clusters is **k=10**

3. The images below show the final image after Means clustering for the cluster values ranging from 2 to 20. As we can also visually see, k=10 seems to have captured all the necessary details of the original image (all distinct colors and lego pattern - four circles - for each color has been captured within these cluster).

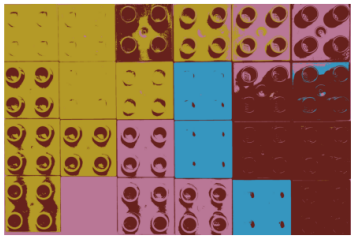
4. k>10 does not add any new information/insight to the final image after clustering but slows down computation.



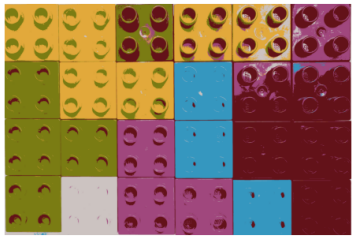
k = 2



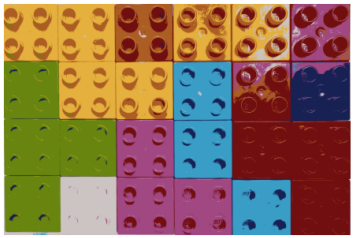
k = 4



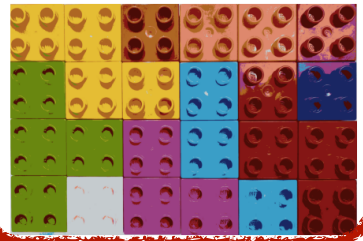
k = 6



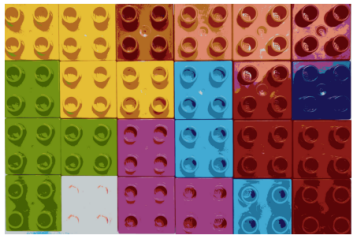
k = 8



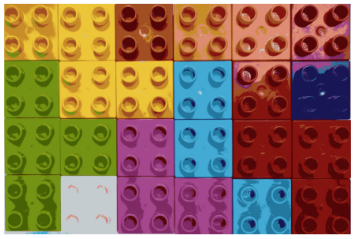
k = 10



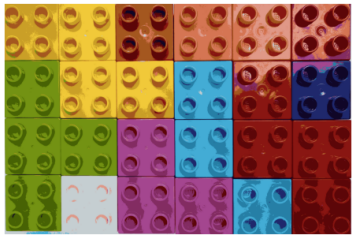
k = 12



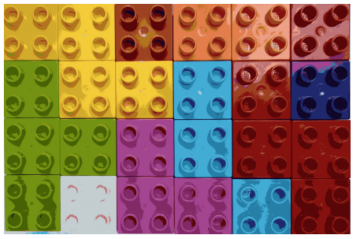
k = 14



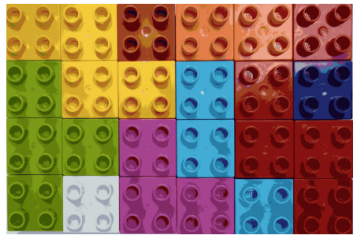
k = 16



k = 18



k = 20



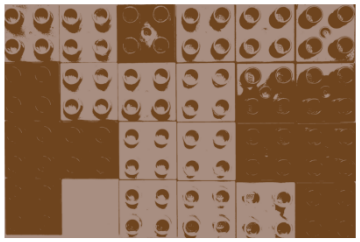
4.2 Gaussian Mixture

1. The Gaussian Mixture algorithm was run for cluster values, k, ranging from 2 to 20.

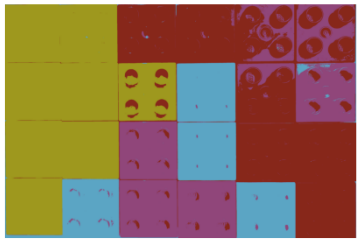
2. The images below show the final image after applying the Gaussian Mixture algorithm for the cluster values ranging from 2 to 20. As we can also visually see, **k=14** seems to have captured all the necessary details of the original image (all distinct colors with most of the lego pattern, except the white piece).

3. k>14 does not add any new information/insight to the final image after clustering but slows down computation.

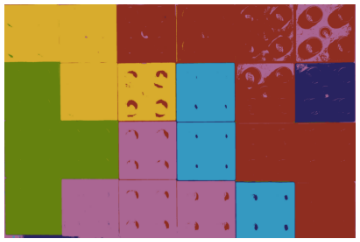
k = 2



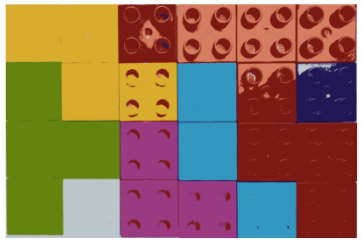
k = 4



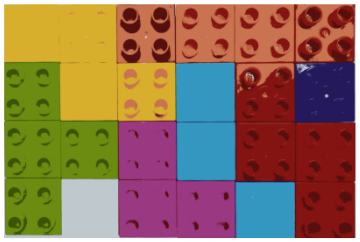
k = 6



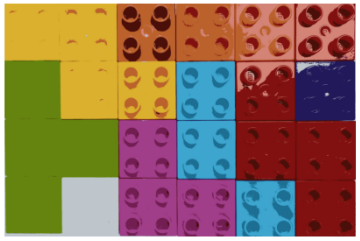
k = 8



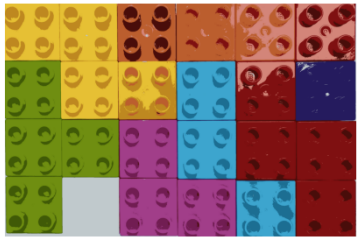
k = 10



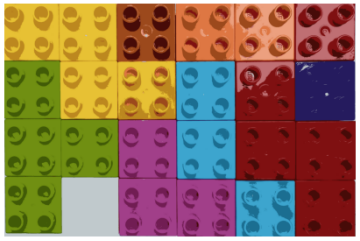
k = 12



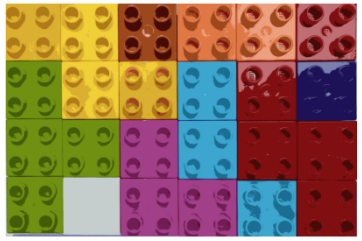
k = 14



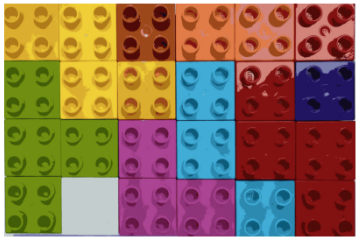
k = 16



k = 18



k = 20



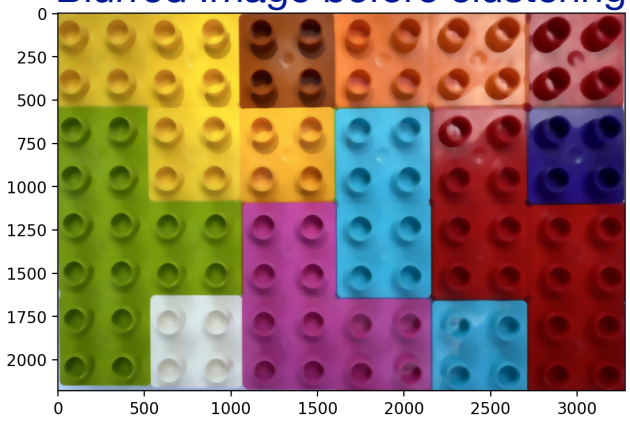
4.3 Mean Shift

1. The Mean Shift algorithm was applied here.

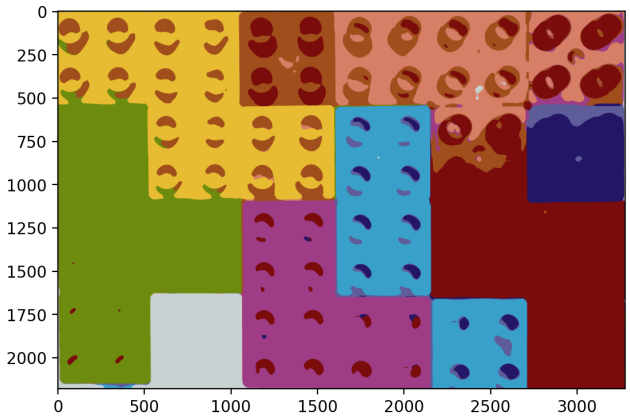
2. The original image was blurred before feeding it to the algorithm. Blurring (or smoothing) is useful to remove any unnecessary details from the original image that might not contribute in segmenting the original image.

3. The images below show the final image after applying the Mean Shift algorithm. The bottom figure indicates that the algorithm did a good job of capturing all the distinct colors of the original image but it, however, hasn't been able to extract the 4-circle detail within each colored piece.

Blurred Image before clustering



Final Image after clustering



4.4 My Pick - KMeans

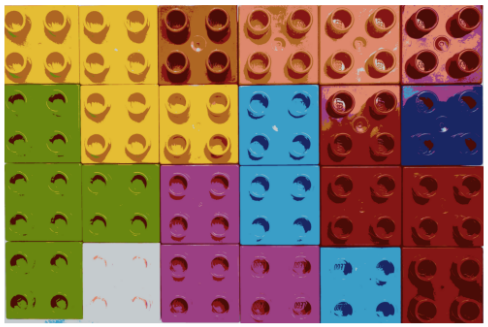
KMeans Clustering algorithm has outperformed the Gaussian Mixture and Mean Shift algorithms for the following reasons:

1. Compared to Gaussian Mixture, Means requires fewer input cluster values (k) to segment the original image without loss of important details or information. At fewer k=10 clusters, KMeans captured all the original colors and all the 4-circle details from each lego piece. The optimum value for number of clusters for Gaussian Mixture was k=14. This also made KMeans a little faster (but not by much) to Gaussian Mixture.

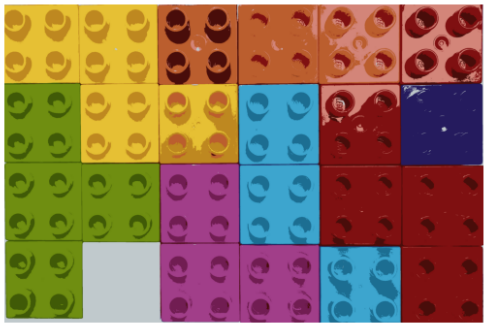
2. Mean Shift, was faster than KMeans in running speed but performed poorly at capturing all the important details and information of the original image.

Original Image

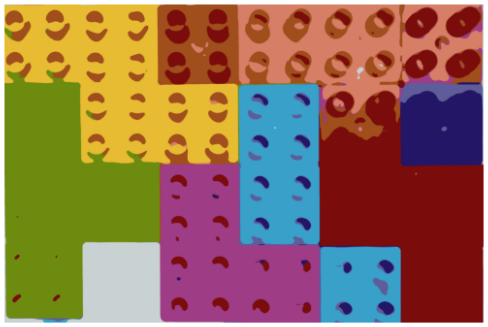
KMeans (k=10)



Gaussian Mixture (k=14)



Mean Shift



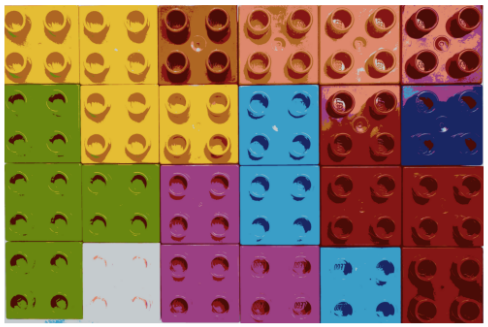
5. Summary

Key findings, Insights and Issues

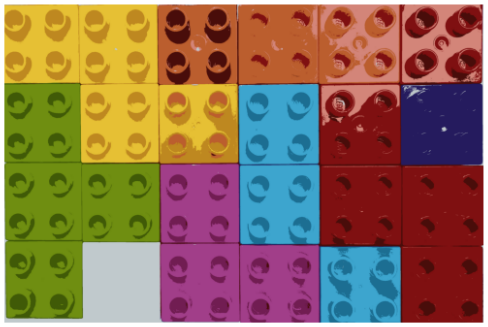
- KMeans captured all the necessary information (colors and details) for k=10. Running speed was modest about 10 minutes.
- Gaussian Mixture captured all the required information (colors and details) for a larger k=14. Slower than KMeans.
- Mean Shift was faster than both KMeans and Gaussian Mixture but was unable to capture all the relevant details like the other two.
- Issue: DBSCAN somehow did not work for this image. Connection to Jupyter notebook kept breaking while fitting with DBSCAN. Could it be due to the high resolution of the image? It is unclear.

Original Image

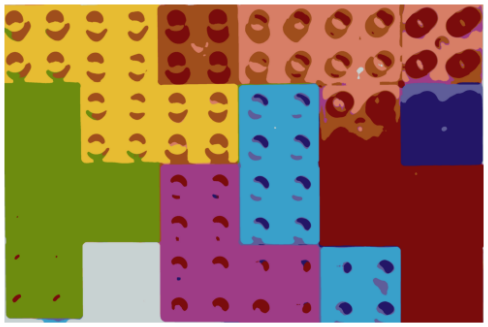
KMeans (k=10)



Gaussian Mixture (k=14)



Mean Shift



6. Next Steps

- Use the segmented image (after clustering) for more advanced machine learning models like deep learning.
- The image used here is very high resolution. It will be interesting to see how reducing the resolution or the size of the image might affect clustering.
- It also be interesting to understand what minimum resolution is needed to segmentation to be useful for deep learning.
- More complicated images can be used to compare the algorithms implemented. It will be interesting to see how the algorithm relate to more complex and detailed images.