CS550 Machine Learning Homework 3 Spring 2022

Kutay Taşcı - 22101359

I. Introduction

In this homework we are going to implement K-Means and Agglomerative Hierarchical Clustering algorithm and then we are going to experiment on these algorithms with different settings. In both algorithms we are going to experiment with different k values. For this task we are going to use an image and cluster the RGB values of the given image.

We are going to analyse our experiments with respect to clustering error, computational time and clustered image. Then lastly we are going to use elbow method for finding the best value of K and we are going to show our results.

II. Part 1

In the first part we are going to implement K-means algorithms and run experiments on our implementation. We are first going to explain our implementation details. Then run experiments on the values of k = { 2, 3, 4, 5, 6}. Then report the expected metrics. Lastly we are going to use elbow method and find the best k value available and run experiments on that value.

A. Implementation Details

For our implementation we used python programming language. For handling numeric operations we used Numpy, for image processing we used Pillow and lastly we used time library to measure execution time.

We have created a K-Means object for ease of use. K-Means object has three methods. First method is initializer method which initializes the object. Second method is the fit method for training the model. Last method is the predict method for using the trained model.

For initializing centroids we randomly choose k points from our data-set (we read every pixel from image and form a data-set). Then we start our training loop which ends when the change in centroids is less than 0.001 or maximum iteration is achieved. In every iteration we cluster the points according to current centroids. Then we set the new centroids as the average of generated clusters until model converges.

In prediction method we assign the given data-points to closest centroid and return both label and the distance. We are going to use distance for calculating clustering error.

B. Experiments

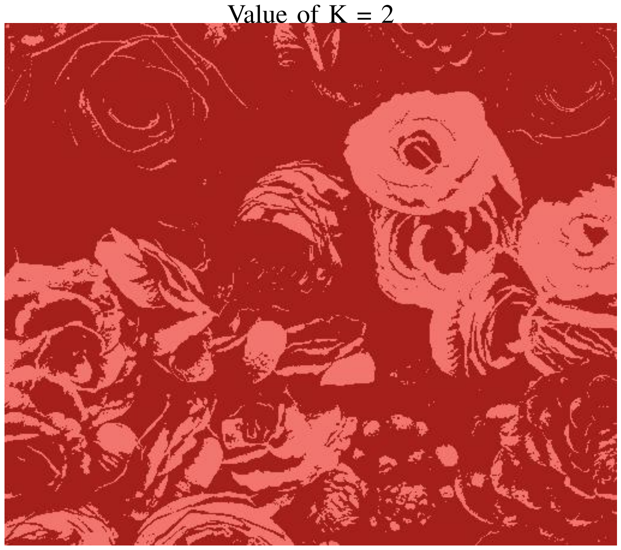
In this section we are going to experiment on our implementation with using different values of K. For each value we are going to give clustering error and running time of the training process.

Bellow you can see the results of our experiments. For evaluating clustering error, it is important note that data is normalized by subtracting mean and dividing by deviation.

Value of K	Training Time(Sec)	Clustering Error(AVG)
2	29.4	0.70
3	10.2	0.52
4	56.4	0.44
5	103.2	0.38
6	79.2	0.36

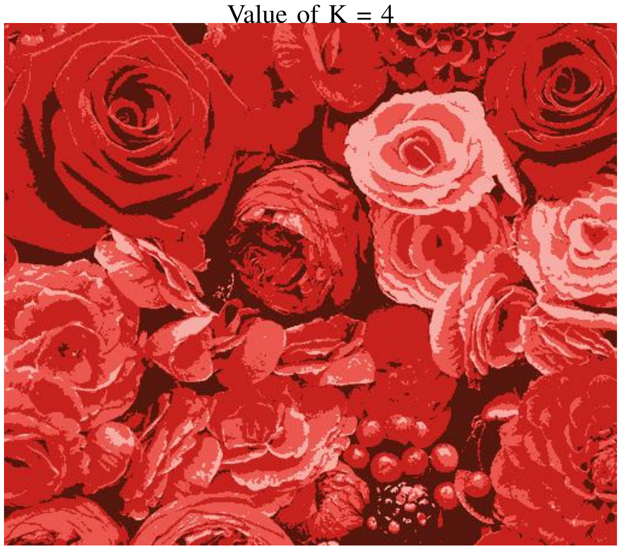
As you can see in the above our clustering error decreased with the value of K. Also our training time is expected to increase with the value of K. However the converging time is trivial since the centroids are initialized randomly training time is highly depends on the initialization. Bellow you can see the clustered images.

Value of K = 2

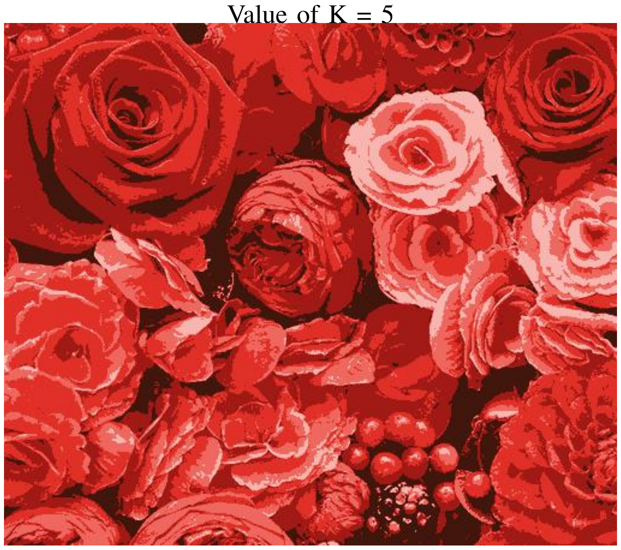


Value of K = 3

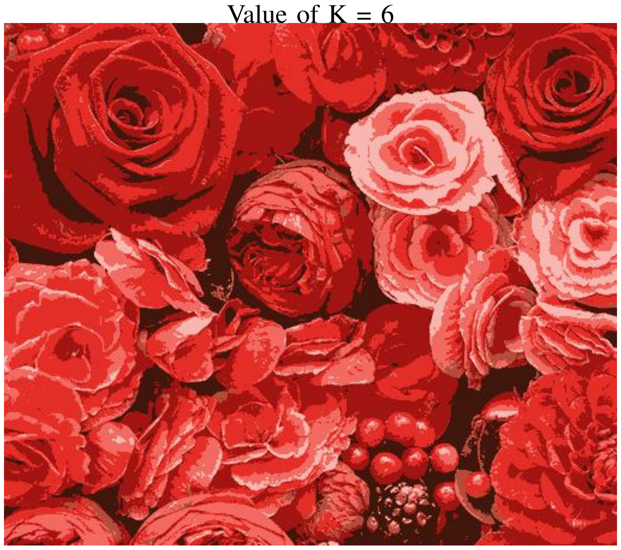
Value of K = 4



Value of K = 5

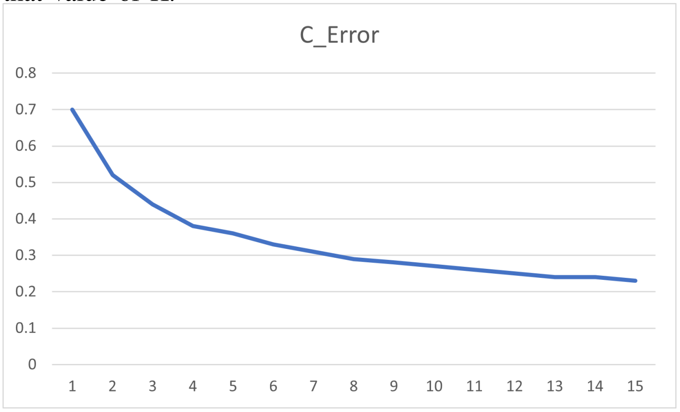


Value of K = 6



C. Model With the Best K Value

In section we are going to find the best value for K and give our results for this Model. For choosing the best value of K we are going to use elbow method. As extra values we are going to try the values of K = {2, 3, 4, 5, 6, 8, 10, 12, 14, 16}. When error decrease is converged we are going yo use that value of K.



As you can see in the plot above clustering error stabilizes after K = 8. So we can say that it is a good value for K. Training time for K = 8 is 179 seconds and average clustering error for all samples is 0.31. Also you can find the clustered image of this model in bellow.

III. Part 2

In the second part we are going to implement Agglomerative Hierarchical Clustering algorithm and run experiments on our implementation. We are first going to explain our implementation details. Then run experiments on the values of k = { 2, 3, 4, 5, 6}. Then report the expected metrics. Additionally in this part we are going to try reducing the training cost of AHC algorithm.Since computational cost of AHC algorithms is too high we need to regularize it somehow. Lastly we are going to use elbow method and find the best k value available and run experiments on that value.

A. Implementation details

For our implementation we used python programming language. For handling numeric operations we used Numpy, for image processing we used Pillow and lastly we used time library to measure execution time.

We have created a AHC object for ease of use. AHC object has three methods. First method is initializer method which initializes the object. Second method is the fit method for training the model. Last method is the predict method for using the trained model.

Our model start with initialing all data-points as a cluster. Then it searches for the closest clusters. In this step for computational efficiency if two clusters are closer than 0.000001 model breaks the search loop. After finding closest clusters. Algorithm takes the average of their centroids and forms a new cluster. Then we remove old centroids and add the new centroid. Therefore our linkage method is ”average”. We continue this process until there are K clusters remaining.

In prediction method we assign the given data-points to closest centroid and return both label and the distance. We are going to use distance for calculating clustering error.

B. Reducing Training Time

AHC algorithms computational cost is much higher than the K-Means algorithms. Because it is not dependent on the value of K. Rather it is dependent on the length of the training data. In this algorithm we used 2 methods for reducing our training time. Since standard algorithm takes too long to execute. We only used high performance models. Because of this there are not any time comparison between normal and high performance model. But it is safe to say that high performance model is faster, because normal models idle time is much longer than high performance models finish time.

First method we used is actually explained in the previous section. In this method when searching for the closest clusters for merging them we speed up the process by simply excepting the clusters close enough. Intuition behind this idea is to eliminate the complexity when dealing with dense clusters.

Second method we used aims to reduce data points directly. As I said in first paragraph models runtime depends on the length of the input. For reducing the data I have tried to bucket the colours, but this resulted shifting in colours. Then, I tried to randomly sample from the data-set. This idea works well but not perfect. Increasing the sampling size makes for a better model. Intuition behind this idea is that uniformly selecting random data from much wider data somehow represents overall data distribution.

C. Experiments

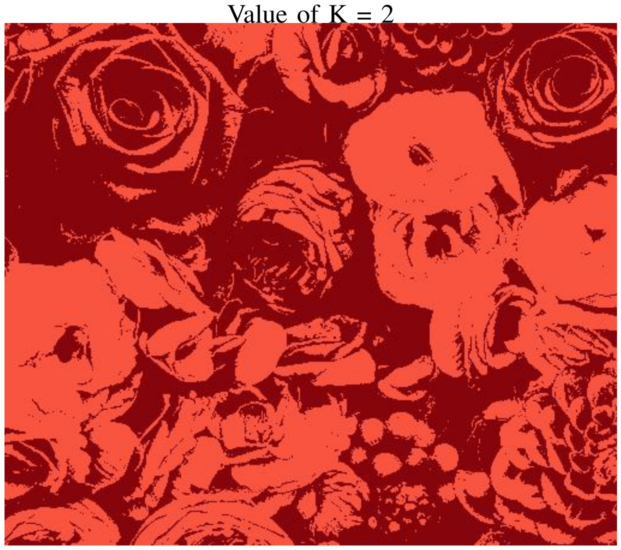
In this section we are going to experiment on our implementation with using different values of K. For each value we are going to give clustering error and running time of the training process.

Bellow you can see the results of our experiments. For evaluating clustering error, it is important note that data is normalized by subtracting mean and dividing by deviation. Also model is trained on subsample of dataset, but evaluated on full dataset. We are going to use sample size 5000 for experiments. Since sampling makes our model stochastic, the error rate and training time is average of 3 models with that k value.

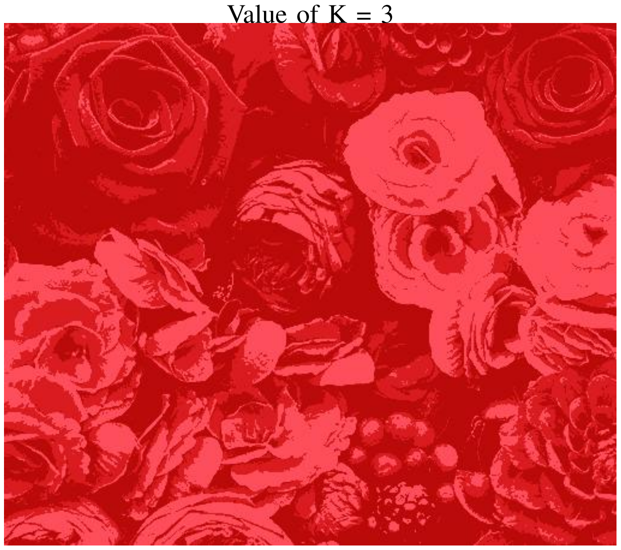
Value of K	Training Time(Sec)	Clustering Error(AVG)
2	42.6	0.69
3	43.1	0.60
4	42.7	0.57
5	42.4	0.45
6	42.5	0.42

As you can see above training is independent from the value of K. For sampling size 5000 it is approximately 42.5 seconds. Also our errors are relatively good. Considering they are average of 3 models. Bellow you can see sample clustered images from our experiments.

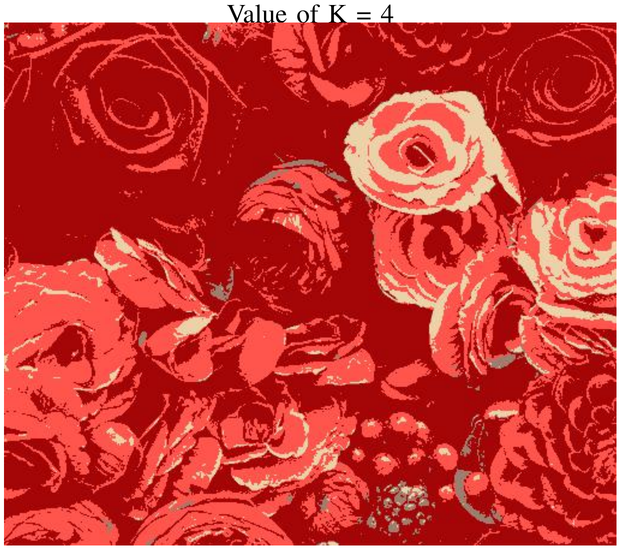
Value of K = 2



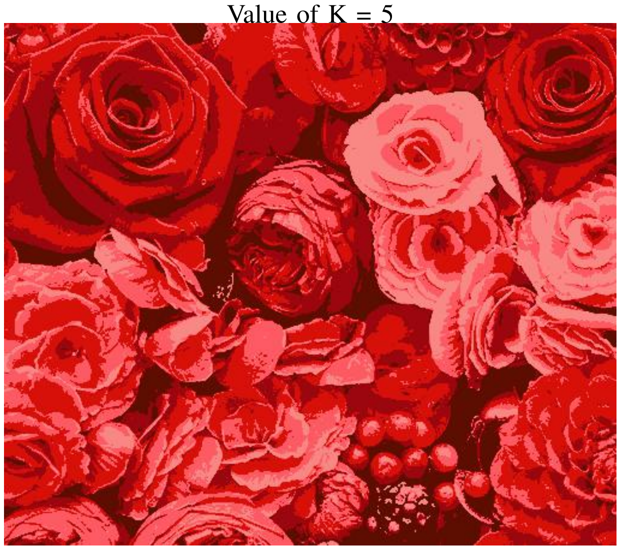
Value of K = 3



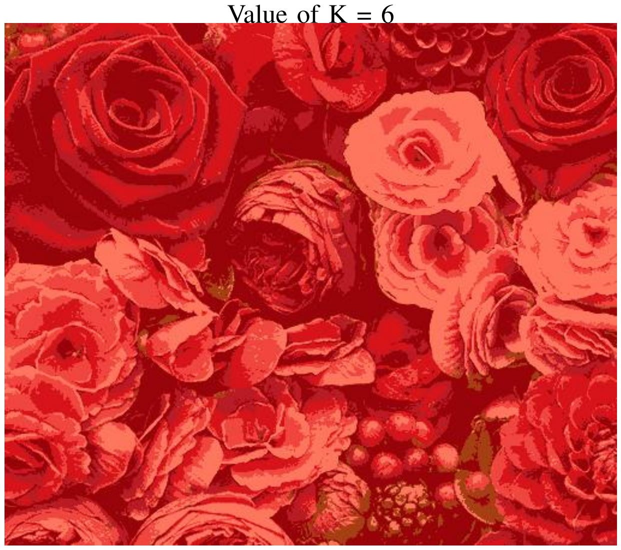
Value of K = 4



Value of K = 5

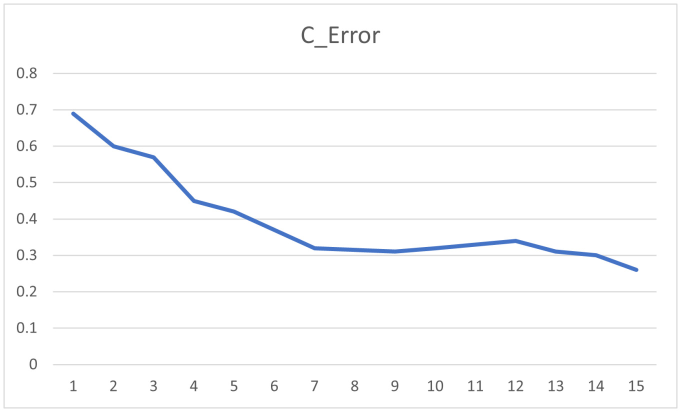


Value of K = 6



D. *Model With the Best K Value*

In section we are going to find the best value for K and give our results for this Model. For choosing the best value of K we are going to use elbow method. As extra values we are going to try the values of K = {2, 3, 4, 5, 6, 8, 10, 12, 14, 16}. When error decrease is converged we are going yo use that value of K. As an extra work that is not in this assignment. We are also going to experiment on sampling size to see the effects of it. For this experiment we are going to use best value of K.



As you can see above my method for reducing running time of the algorithm caused stochastic behaviour in error.K = 8 is again the best value for K according to this experiment. With the average clustering error of 0.32 and training time of 42.4. Also you can see the clustered image bellow.

E. Effects of Sampling

Additionally we are going to analyse the effects of the random sampling. Random sampling speeds up the training process significantly but also makes the results stochastic. Depending on the initial data our results on overall data changes. In this section we are going to give insights about its effects. For this task we are going to use K=12.

Sample Size	Training Time(Sec)	Clustering Error(AVG)
3000	15.6	0.31
5000	42.1	0.30
7000	82.8	0.27

As expected training time increased with the sample size. As the sample size increased clustering error became much stable (As an subjective observation). Also increased in some level. Some experiments with 3000 sample size is also performed well but some of them performed much worse.

IV. Conclusion

In this assignment we implemented 2 clustering algorithms. First we implemented K-Means algorithm and experimented on our implementation. Then we implemented AHC algorithm and tried to increase the efficiency of the algorithm. After analyzing our results we can see that both algorithms has their strength and drawbacks.

Strength of K-means algorithms is that it is efficient and gives stable results. However complexity of the model increases with the value of K. On the other hand AHC performs in same complexity for different values of K, but generally has more training time than K-Means. That makes the AHC more suitable for large values of K.