Homework Assignment 04

Due: Nov 20, 2023

**1. Clustering with the k-Means algorithm.**

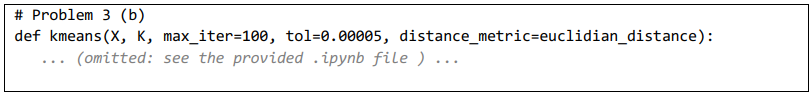
(a) (4 pt.) In your report, write down the definition of the Euclidean distance between two points a and b. Go to the Jupyter notebook and write function euclidian\_distance(a, b) that takes two vectors and returns the Euclidean distance between them.

Solution)

The Euclidean distance between two points is defined as the square root of the sum of the squared differences between corresponding coordinates.



(b) (3 pt.) Find the code cell starting with “# Problem 1 (b)” in the provided Jupyter notebook. It is a completely functioning implementation of the k-means algorithm (you do not need to modify the code in the cell for addressing problems 1(b)-1(f)). Spend some time with the code and try to understand its logical flow. Recall our discussions in class.



\* Disclaimer: The provided code is written for ITP40010 and should be used for educational purpose only. The user may experience the division by zero errors when the algorithm yields empty clusters (clusters with 0 instance) during its process.



Regarding the implementation, answer the following questions (include your answer in the report):

- What do you need to provide for ‘X’ and ‘K’?

X: This is the input data for the K-means algorithm, represented as an array or matrix.

K : This represents the number of clusters. This determines the number of centroids to be created and the number of clusters to which data points will be grouped.

- What do the values ‘max\_iter’ and ‘tol’ do?

max\_iter: This is the maximum number of iterations that the K-means algorithm will perform.

tol: It is the convergence threshold. If the change in sum of squares of error between successive iterations is less than this tolerance value, the algorithm stops.

- What do the output ‘c’ and ‘centroids’ contain?

c: It is an array containing the cluster assignments(cluster number) for each data point. Each value of 'c' represents the cluster to which that data point belongs.

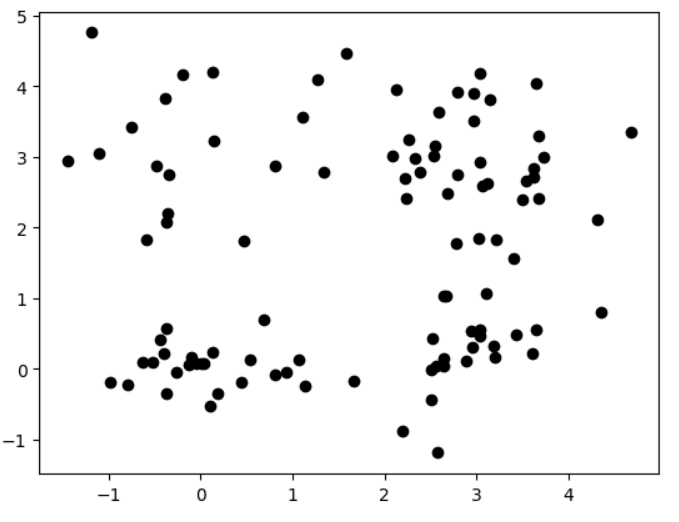
centroids: It is an array containing the final centroids of the clusters. Each row corresponds to a centroid.

- What do the output ‘log\_centroids’, ‘log\_c’, and ‘log\_sse’ contain?

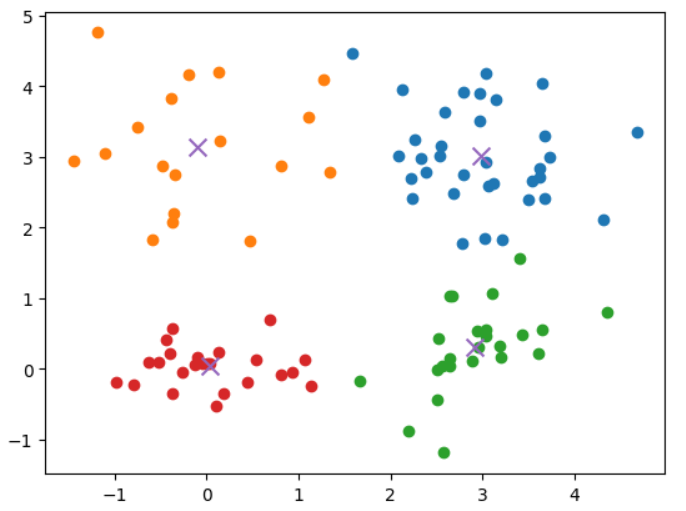
log\_centroids: This stores the centroids at each iteration during algorithm execution.

log\_c: This stores the cluster assignments at each iteration during algorithm execution. log\_sse: This stores the sum of squared errors (sse) at each iteration during algorithm execution.

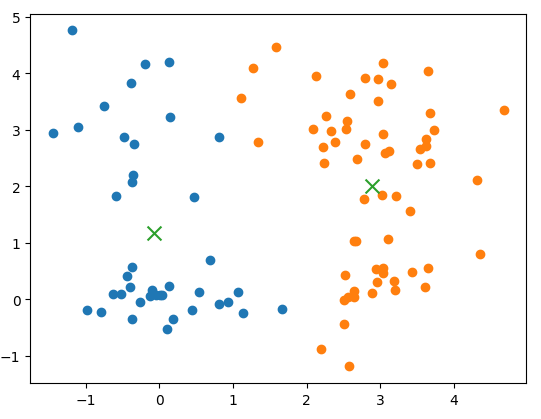
(c) (6 pt.) Find the code cell starting with “# Problem 1 (c)” in the provided Jupyter notebook. Function “generate\_random\_data(N)” is a utility that generates a 2-dimensional synthetic dataset (an artificial dataset with 2 features). The last two lines of the code cell invoke this function and visualize the generated dataset by drawing a scatter plot. Take a moment with the code to understand its logical flow. Try to execute and investigate the output of the function (you do not need to modify the code in the cell).



- Once you have reviewed the code, find the cell starting with “# Problem 1 (c) - part 1” and write a code snippet that executes the above k-means clustering implementation on the generated dataset, with k = 4. Visualize the datapoints with colors, representing clusters. Also, indicate the position of the centroids that you obtained after running the implementation. In your report, include the plot that you have created. In the Jupyter notebook, submit your code snippet.



- Now let us move onto the cell starting with “# Problem 1 (c) - part 2” and write a code snippet that clusters the data with k = 2. Visualize the datapoints with colors, representing clusters. Also indicate the position of the centroids that you obtained after running the implementation. In your report, include the plot that you have created. In the Jupyter notebook, include your script.



- Based on the results, between k = 4 and k = 2, which parameter value do you think better and why? Submit your answer in the report. In the Jupyter notebook, include your script.

I think k=4 is better than k=2. We can check that Data pointers are scattered in a nearly square frame. If k is 2, it is divided into horizontal or vertical centers and clustered. As a result, data that is very far from the center value are also closer than the other remaining center, so they are grouped into one group. Even within one group, there is a large difference between the center value and the distance between the data points, and the distance between the data points is also large. However, if k is 4, the distance between the center value and the data is similar around the center value. Therefore, I think k=4 is better than k=2.

(d) (2 pt.) Turn to the code cell starting with “# Problem 1 (d)”. Execute the code cell. Make sure to place the data file (“Mall\_Customers.csv”) in the right location (in the same directory as the .ipynb file is placed) so that the code cell loads the data without any problem. You are given a new dataset containing customer information at a mall1 . The dataset consists of four columns; gender (male/female), age (18-70), annual income (unit: 1000 USD), and spending score (1-100).

In your report, write down your answers to the following questions.

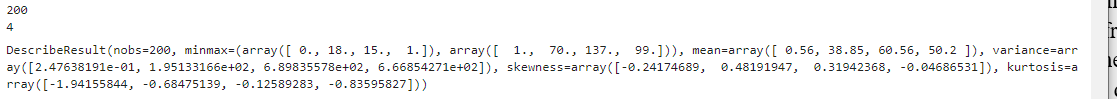
- What does stats.describe(my\_data) do? What kind of information can you find out from the output of stats.describe(my\_data) mean?

Solution)

stat.describ(my\_data) provides a descriptive statistical summary of a dataset. In the output of stats.description(my\_data), I can know information about nobs(number of observations), minmax(Minimum and maximum values for each column(feature) in your dataset), mean(Mean (average) values for each column), variance(variance for each column), skewness, and kurtosis.

As you may notice from the output of stats.describe(my\_data), the dataset consists of the features that have varying ranges from each other. In your report, include a formula that you can use to normalize (standardize) the data. Then, apply the provided function “normalize(X)” to my\_data and examine the resulting dataset with stats.describe(normalize(my\_data)). How does the dataset change after applying normalize()? Describe the differences in your report.

Before normalization

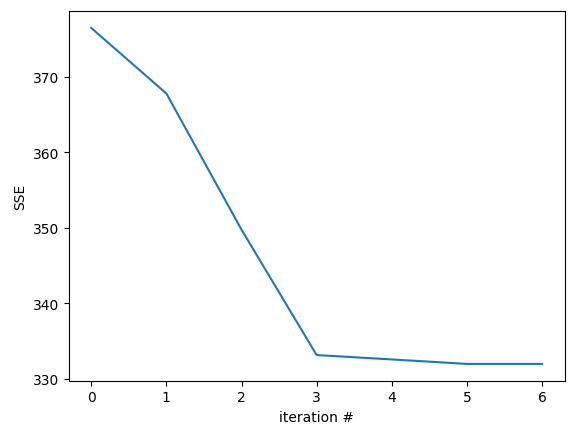


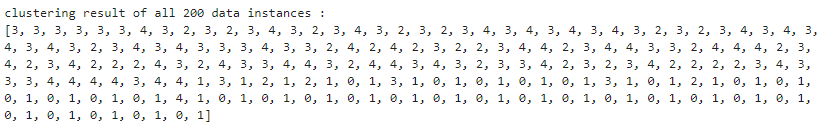
After normalization



The minmax and mean values of the data and the variance values was changed. Through normalization, the data was changed so that the average was almost zero and the variable value was almost one. That's why the value of minmax has also changed.

(e) (4 pt.) Write code that normalizes my\_data and then conduct clustering on it. Use k = 5. In your Jupyter notebook, include a code snippet that performs the task and prints out the clustering result (the cluster membership of all 200 data instances). In your report, include a plot that shows the trace of SSE (sum of squared errors) throughout the clustering (i.e., a line plot of SSE over the iteration number). Explain how SSE changes over iterations.



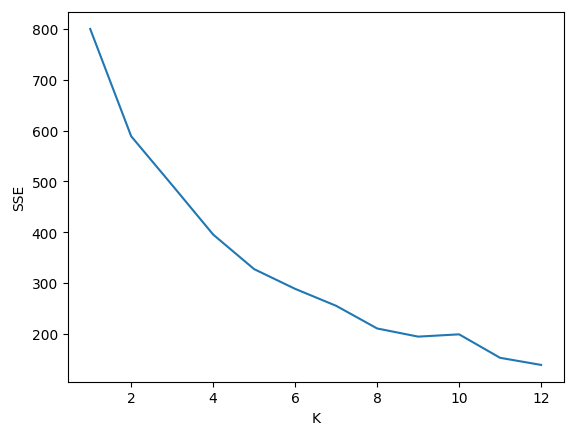


The SSE represents the sum of the squared distances between each data point and its assigned cluster centroid. The K-Means algorithm minimizes SSE by iteratively updating cluster groups and centroids. Assign each data point to the cluster whose centroid is closest to it. Then recalculate the centroids and calculate the SSE using the updated centroids.

If the SSE is decreasing steadily, it indicates that the algorithm is successfully finding better clustering and centroids. In the graph above, when the iteration number became 3, the value of SSE decreased sharply, and after that, it can be seen that the value of SSE does not change significantly.

(f) (Max 10 pt.) Write code that tries out clustering with multiple values of k, ranging between 1 and 12. You may see a runtime warning message “RuntimeWarning: invalid value encountered in true\_divide” while executing. Repeat the code run, until you do not see any warning message.

- (3 out of 10 pt.) In your report, draw and submit an SSE vs K plot that looks similar to the one provided below. In your Jupyter notebook, include your code snippet together with the execution result.



- (Max 2 out of 10 pt.) Which k do you think the best k for this dataset? Write and justify your answer in your report.

I think 8 is the best k for this dataset. This is because, looking at the graph above, the value of SSE decreased most rapidly when K was 8. After that, the SSE value does not change rapidly.

- (Max 5 out of 10 pt.) Perform a formal analysis of the results. You may want to examine all individual clusters for your choice of k, by evaluating the mean, standard deviation, median, minimum, maximum, etc. to understand the properties of each cluster. With your best of knowledge, characterize and distinguish the clusters. (Write your answer in the report.)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Mean | Std | Median | Min | Max |
| 1 | -0.08258198608968023 | 0.973466935206334 | -0.2503914553668266 | -1.7008297638941756 | 1.8944921627227171 |
| 2 | -0.1370074724729699 | 1.0834462469329038 | -0.2503914553668266 | -1.8323776655842676 | 2.235532383381568 |
| 3 | 0.18644883335740234 | 1.08069211010564 | 0.20234083766810235 | -1.1281521496355336 | 2.9176711658902774 |
| 4 | -0.39632854556149005 | 1.0874926727850487 | -1.0610604008183242 | -1.9100207870073298 | 2.9176711658902774 |
| 5 | 0.39295832094536903 | 0.8324322436523648 | 0.6656748447557095 | -1.4963354797273096 | 2.497807445000782 |
| 6 | 0.3875304721356087 | 0.7478060475716023 | 0.5321491942089316 | -1.7547345441612054 | 2.0919990040312264 |
| 7 | -0.3927722547238443 | 1.0084057475164725 | -0.6483200638825694 | -1.7547345441612054 | 1.3743321072795194 |
| 8 | -0.6062982702248846 | 0.7871830755424567 | -0.9580853071635292 | -1.7389991930659479 | 1.6227412377419996 |

The table above shows the calculated values ​​of mean, std, median, min, and max for each cluster. I can see that the data is located between min max values.

(g) (Max 6 extra pt.) While you are tackling the above problems, you may have seen the runtime warning: “invalid value encountered in true\_divide” multiple times. This has occurred due to a particular reason. In your report, explain why this is happening and how the outcome would be like (hint: the disclaimer given in Problem 1(b)). In the Jupyter notebook, modify function kmeans() such that you do not see the warning anymore and obtain correct results.

The runtime warning: “invalid value encountered in true\_divide" is caused by division by zero when calculating the centroids(tmp\_sum / tmp\_count). This can happen when a cluster becomes empty. This mean is sum clusters has zero instance assigned to it, and the code attempts to divide by the count of instances in that cluster.

I changed the code like this:

if tmp\_count == 0:

centroids[k] = prev\_centroids[k]

else:

centroids[k] = tmp\_sum / tmp\_count

The line `centroids[k] = prev\_centroids[k]` mean that if the cluster is empty, the centeroid is set to be the same as the previous centeroid. This prevents denominators from becoming zero in the tmp\_sum / tmp\_count line. If the centeroid is set to 0 or any other value when the value of tmp\_count is 0, the position of the centeroid can change significantly, so I used the previous centeroid.

**2. Image segmentation using k-means.**

(11 pt.) Using the k-means implementation from Problem 1, write a code that segments provided image file hyoam.jpg with k=3, 6, and 9, respectively. Explain the differences among the results with different k in your report. Submit your complete code in the Jupyter notebook. The resulting image would look like:

In the notebook, you are provided with a code snippet that reads in hyoam.jpg, coverts the pixel representation to the RGB color, and reshapes the pixels into a matrix form (X\_rgb; where each row represents a pixel, and each column represents R, G, and B, respectively).

