Caitlyn Jones

Prof. Thomas Fai

MATH 122A: Numerical Methods and Big Data

3 December 2024

Homework 8

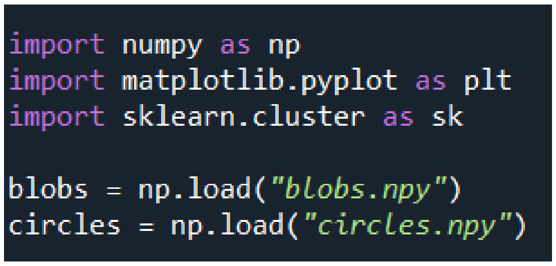
1.

   a. Caitlyn Jones, Jonah Gardenswartz, Eric Huang

   b. For our project, we decided to do Option B: Regression with Clustering. In part B, we will be working with a dataset filled with a list of 44,000 patients. The list will contain the symptoms of the patients and the medical diagnosis they received: Covid-19, flu, cold, or allergies. With the given list, we will separate the dataset into a test set and a sample set. After we have clustered the data, we will use the sample set to predict what medical diagnosis they will receive. Our goal is to be able to accurately predict the diagnosis of a patient from the test set.

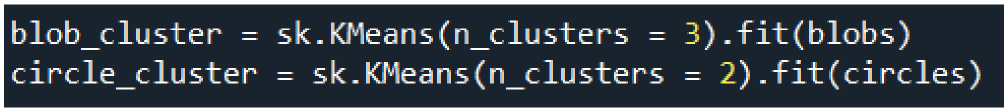
2.

   a. To begin, I loaded the two data sets, `blobs.npy` and `circles.npy`.



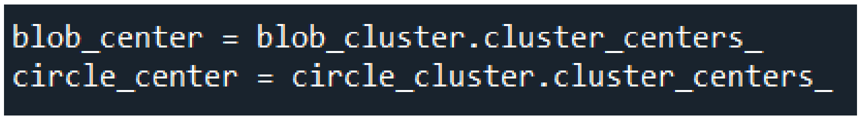
```
import numpy as np
import matplotlib.pyplot as plt
import sklearn.cluster as sk

blobs = np.load("blobs.npy")
circles = np.load("circles.npy")
```

Then, I performed k-means clustering, using 3 clusters for the blobs and 2 clusters for the circles.



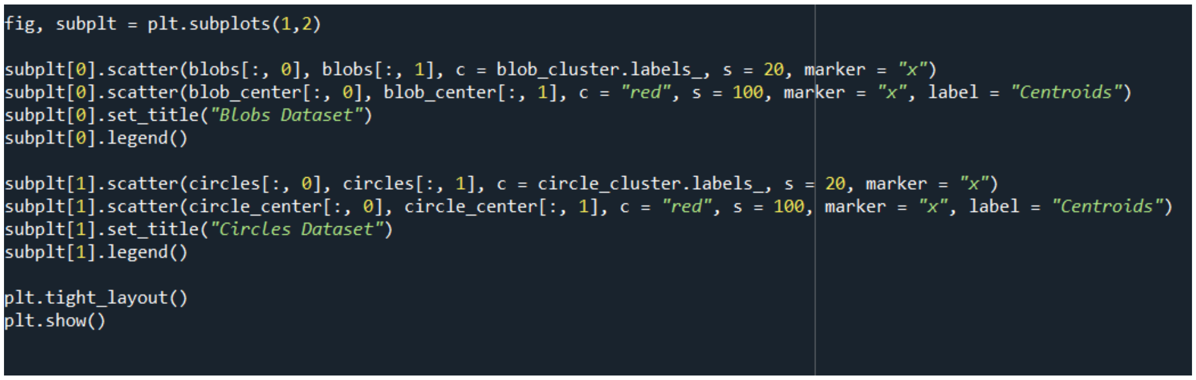
```
blob_cluster = sk.KMeans(n_clusters = 3).fit(blobs)
circle_cluster = sk.KMeans(n_clusters = 2).fit(circles)
```

b. Then, I plotted the datasets along with their cluster cent roids. To do so, I first created variables to hold the centroids.

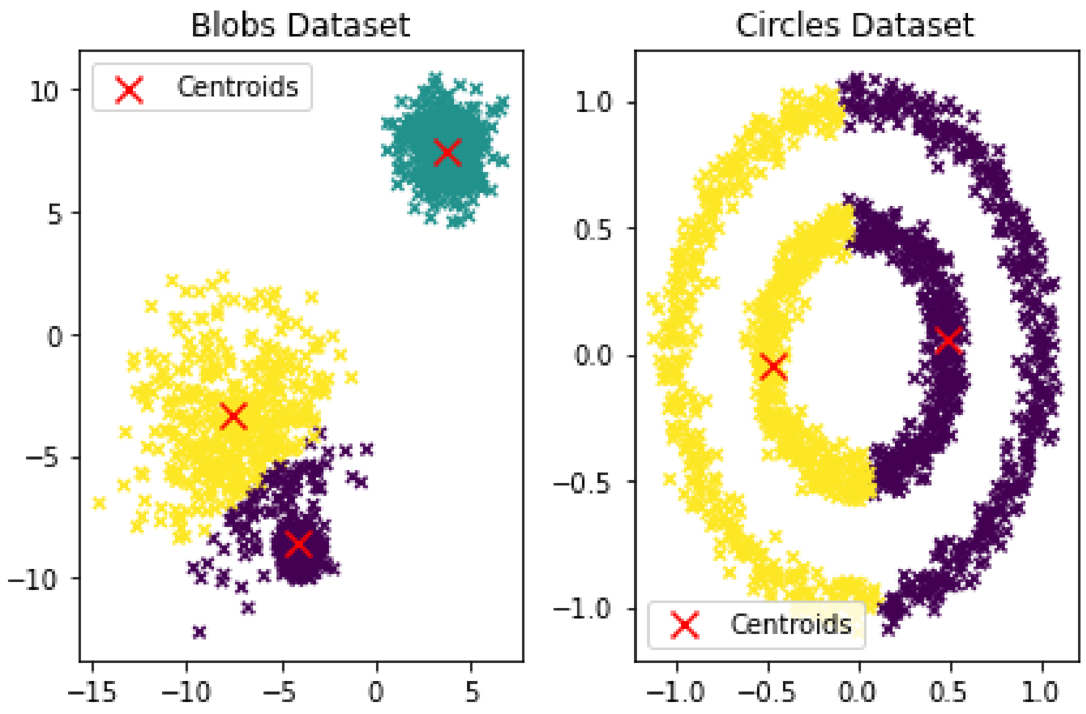


```
blob_center = blob_cluster.cluster_centers_
circle_center = circle_cluster.cluster_centers_
```

Then, I created plots of the blobs and the circles, with various colors to demonstrate the various points. Included on these graphs are the highlighted cluster centroids.

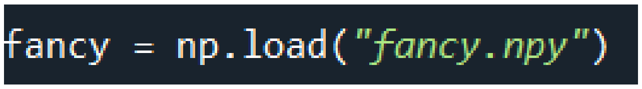


```
fig, subplt = plt.subplots(1,2)

subplt[0].scatter(blobs[:, 0], blobs[:, 1], c = blob_cluster.labels_, s = 20, marker = "x")
subplt[0].scatter(blob_center[:, 0], blob_center[:, 1], c = "red", s = 100, marker = "x", label = "Centroids")
subplt[0].set_title("Blobs Dataset")
subplt[0].legend()

subplt[1].scatter(circles[:, 0], circles[:, 1], c = circle_cluster.labels_, s = 20, marker = "x")
subplt[1].scatter(circle_center[:, 0], circle_center[:, 1], c = "red", s = 100, marker = "x", label = "Centroids")
subplt[1].set_title("Circles Dataset")
subplt[1].legend()

plt.tight_layout()
plt.show()
```



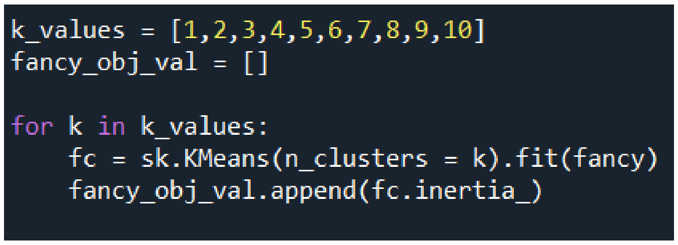
As you can see, for the blob dataset, the k-means clustering accurately clusters the data into 3 groups. Furthermore, each centroid is located at the center of each centroid, which emphasizes the accuracy of the data's structure and its clustering. However, in the circle dataset, the k-means clustering is not as accurate, due to the fact that the clusters are non-linear. Instead of the clusters being determined by the circular structure of the data set, they are based on the linear proximity. Furthermore, the centroids are not aligned with the geometry of the data, but instead are aligned at the center of each inaccurately determined cluster.

c. First, I loaded the new dataset into my code.



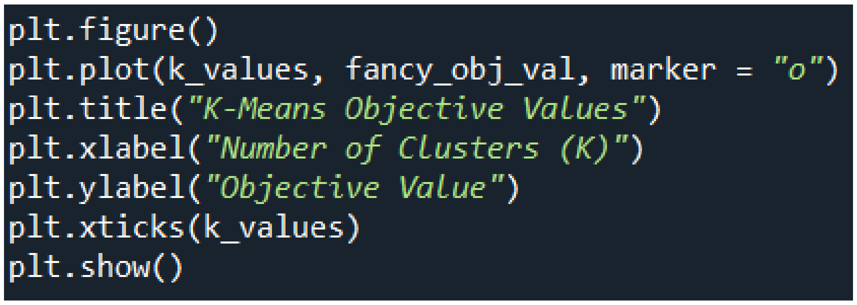
```
fancy = np.load("fancy.npy")
```

Then, I performed k-means clustering with k values from 1-10 and computed the objective value for each cluster.

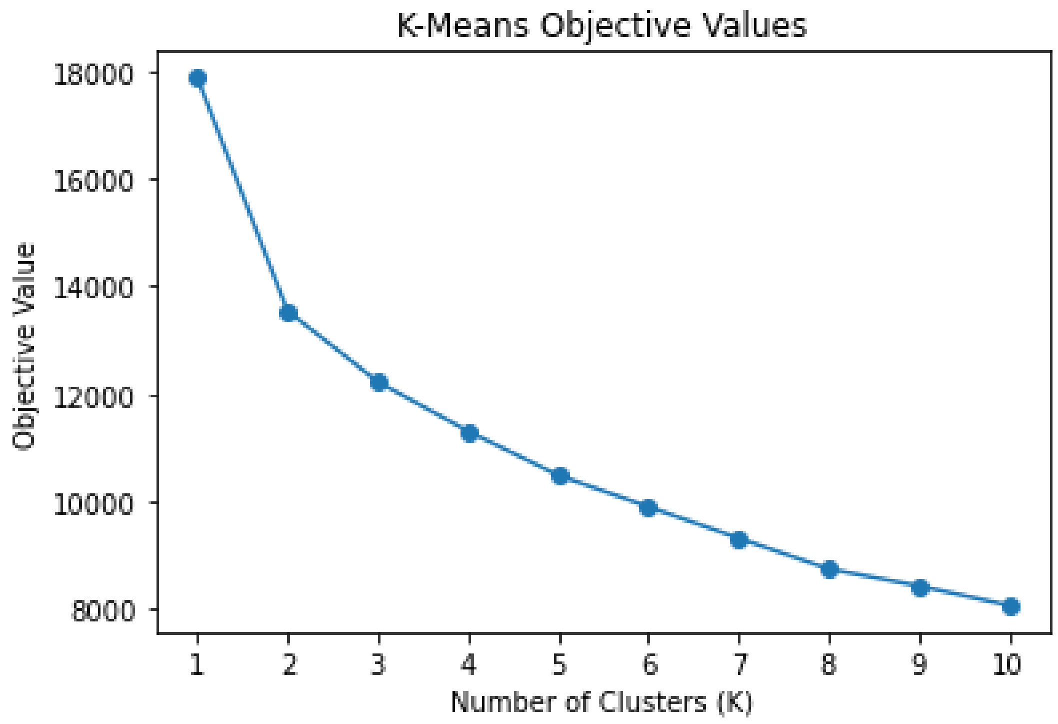


```
k_values = [1,2,3,4,5,6,7,8,9,10]
fancy_obj_val = []

for k in k_values:
    fc = sk.KMeans(n_clusters = k).fit(fancy)
    fancy_obj_val.append(fc.inertia_)
```

Then, I plotted the objective values on a graph.



```
plt.figure()
plt.plot(k_values, fancy_obj_val, marker = "o")
plt.title("K-Means Objective Values")
plt.xlabel("Number of Clusters (K)")
plt.ylabel("Objective Value")
plt.xticks(k_values)
plt.show()
```



As you can see, the reasonable choice for k would be 2, as that is the first point where you can see a noticeable flatten on the curve. This correlates to the point in which the rate of decrease in the objective values slow down, which means that the error is balanced while minimizing the number of clusters.