CS 584: Data Mining (HW4)

Md. Ridwan Hossain Talukder
miner2 ID: Luffy
HW4-Iris: V-Score: 0.95 and Rank 26
HW4-Image: V-Score: 0.79 and Rank 24

I. Problem Statement

There are two parts (Part 1: HW-Iris and Part 2: HW-Image) to this assignment. We have to implement a K-Means algorithm and test the accuracy of the algorithm in Part 1 and use the algorithm to solve a more detailed implementation of the clustering problem in Image data.

II. K-means Implementation

K-means is a pretty straightforward algorithm. It follows the steps below:

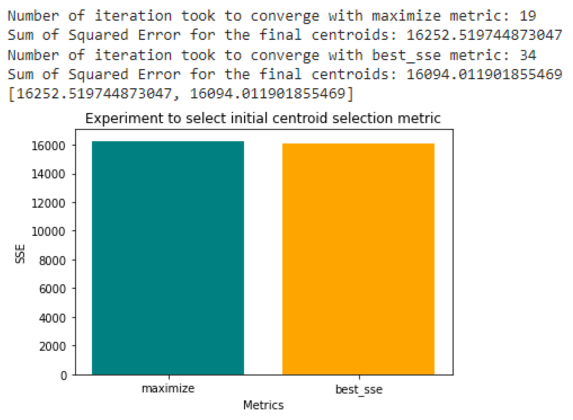
- Step 1: Pick the initial k number of centroids
- Step 2: Cluster the data points around the centroids based on a distance metric
- Step 3: Re-calculate the new centroids from the clusters
- Step 4: Keep repeating Step 2 and Step 3 until convergence

So there are three major challenges, first, how do we calculate the initial centroids, second how to find which data points belong to which cluster, and finally what are the criteria of convergence?

A. Calculating the initial center points or centroids

I have taken two different approaches to selecting the initial centroids. And after running some experiments using both approaches, the approach that gave the lowest sum of squared error (SSE), my internal evaluation metric to evaluate the performance, I decided to choose that to do my final prediction. These two approaches are:

1) Maximize Distance Method: For this approach, I randomly select a point from the data points and it acts as the first centroid of my k number of centroids. Then I calculate the distance of all data points from the first centroid and select the data point that has the maximum distance from the first centroid. Now for selecting the rest of the centroids (3rd to k), I calculate the distance of the remaining data points (except the already chosen centroids) from the selected centroids and select the point that is almost equally distant from the current centroids. Suppose we have already selected [4,20,12] as the centroids and the candidate for the 4th centroids are [2, 18, 8, 14]. In this case, my algorithm will choose 8 as it is more equally distant from the current centroids than the other candidate points. And thus after a k-2 number of iterations, we will find our initial k number of centroids.



```
Number of iteration took to converge with maximize metric: 19
Sum of Squared Error for the final centroids: 16252.519744873047
Number of iteration took to converge with best_sse metric: 34
Sum of Squared Error for the final centroids: 16094.011901855469
[16252.519744873047, 16094.011901855469]
```

Fig. 1. Experiment to select the initial centroid approach

2) Random Centroids with Lowest SSE: For this approach, I randomly choose k number of data points from the dataset and find the sum of square error for the selected centroid. And repeat the process 100 times to select the centroids with the lowest SSE and select them as our initial centroids.

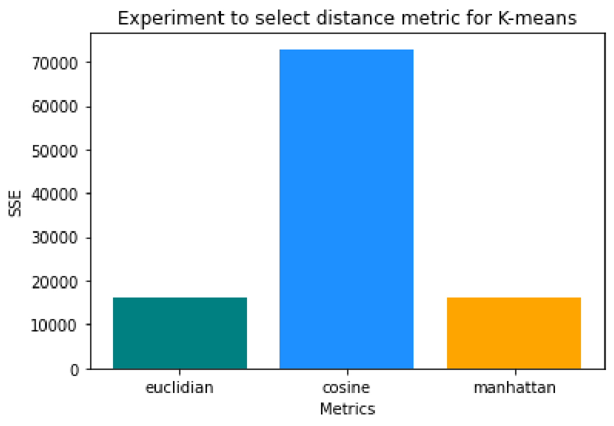


Fig. 2. Experiment to select the distance metric

Both these approaches performed almost equally well in the experiments and for the Image Data sets. But for the iris dataset, Random Centroids with Lowest SSE performed way better(0.95 V-Score in Miner) compared to Maximize Distance Method (0.86), though the number of iterations took to converge and SSE was lower for Maximize Distance Method (20 iterations, 15742.73 SSE for iris data,) than Random Centroids with Lowest SSE (55 iterations, 18962.238 SSE for image). So for the final prediction in both the data set and as the default method for my K-means algorithm, I used Random Centroids with Lowest SSE method.

B. Calculating the clusters

For calculating the clusters, I used Euclidean Distance as my distance metric as the other two metrics that I used (Cosine Distance and Manhattan Distance) did not perform well in the iris data set and also not in my experiments. Euclidean distance metric produced the lowest SSE for the same setup of the experiment. Shown in Figure 2

C. Convergence

My K-means converged when the centroids in two consecutive runs remains the same, or if the number of iterations exceeds the maximum iteration, by default which is 500. But the users can change it to any number they want.

III. K-MEANS CLUSTERING ELBOW METHOD AND SSE PLOT

For the internal evaluation metric, I choose the Sum of Squared Error(SSE) and below I plot this metric on the y-axis (Figure 3 for Image data and Figure 5 for Iris data set) with the value of K increasing from 2 to 20 in steps of 2 for the Image data and 2 to 10 in steps of 1 for the Iris data. And as we can see that the Elbow point (the point after which the SSE or inertia starts decreasing in a linear fashion)) for image data is in 10 while for iris data it lies in 3, which means that the number of clusters we are using for validation is the optimal number of clusters and is supported by this graph.

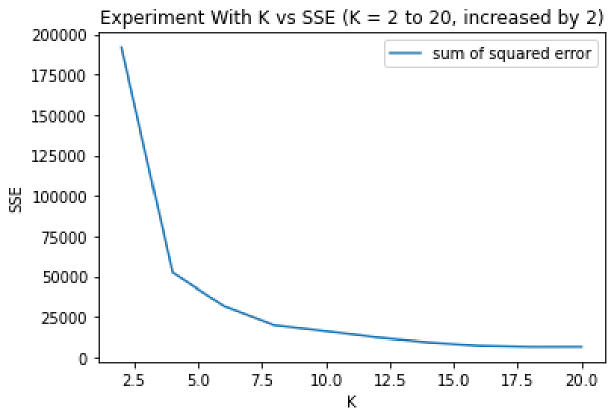


Fig. 3. K vs SSE (Image Data)

IV. FEATURE REDUCTION

A. Iris Dataset

Iris dataset is a famous dataset with a very lower dimension of 4, yet to make a better clustering I first studied the correlation between the features and found out that feature a is highly positively correlated with features c and d so we dropped it along with b which has a low negative correlation

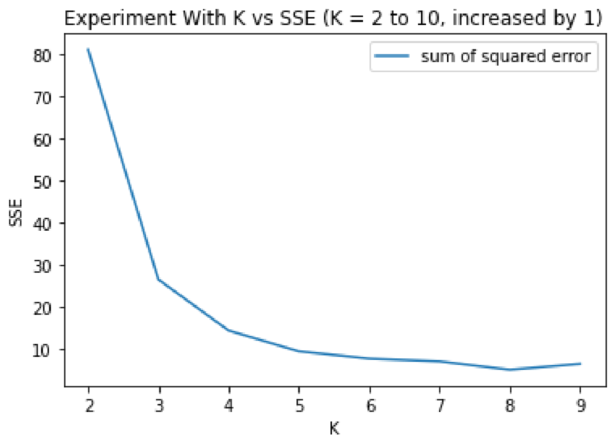


Fig. 4. K vs SSE (Iris Data)

which gave a 91% accuracy on miner2. Then I used Principal component analysis with n_components = 1, which is a linear dimensionality reduction technique to project the data to a lower dimensional space. And it did a better performance with a 95% accuracy on miner2. This shows that the K means algorithm is performing well.

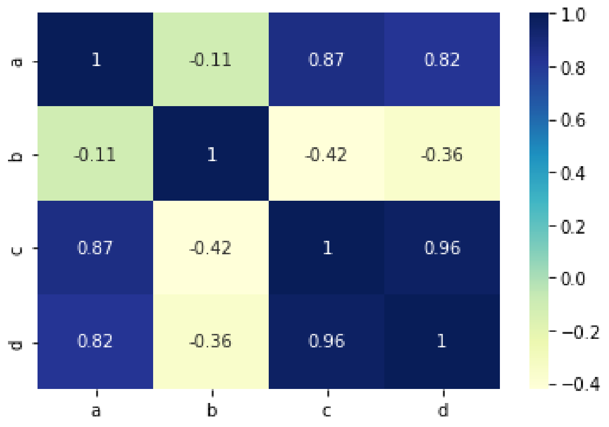


Fig. 5. Correlation in Iris Dataset

B. Image Dataset

Compared to the iris dataset, the image dataset has a higher dimension (784) and I applied some of the feature reduction techniques before using the K-Means algorithm for clustering the data points. First I used two well-known Linear dimensionality reduction techniques PCA and Truncated_SVD. These methods can be applied to linear data and generally do not perform well on non-linear data. And in Figure 6 we can see that using only PCA or Truncated_SVD on the image datapoints were not that helpful as the points are vastly scattered and the SSE is way too higher. So it did not do well in the image dataset. So I then used some of the non-linear methods (Manifold learning) eg: t-distributed Stochastic Neighbor Embedding (t-SNE), Isomap, MDS, and

Uniform Manifold Approximation and Projection (UMAP) but as Isomap and MDS took a huge time compared to the other two methods, I decided to stick to only t-SNE and UMAP. t-SNE focuses on the local structure of the data and tends to extract clustered local groupings of samples. It may be useful to be able to classify samples based on the local structure to visually comprehend a dataset that comprises numerous manifolds simultaneously, as is the case with the digits dataset. The Scikit-learn documentation recommends us to use PCA or Truncated SVD before t-SNE if the number of features in the dataset is more than 50, so I used it with PCA and it did perform way better than PCA or Truncated_SVD. But UMAP is a more recently developed non-linear dimensionality reduction method that performed better than t-SNE in many cases, also for this dataset it had a very low SSE compared to t-SNE and as it can be used with the Scikit-learn structures I used it with n_component=2 and got a better cluster than t-SNE with lower SSE. UMAP is a fairly flexible non-linear dimension reduction algorithm. It seeks to learn the manifold structure of the data and find a low-dimensional embedding that preserves the essential topological structure of that manifold. Which is why it is scalable and fast to use.

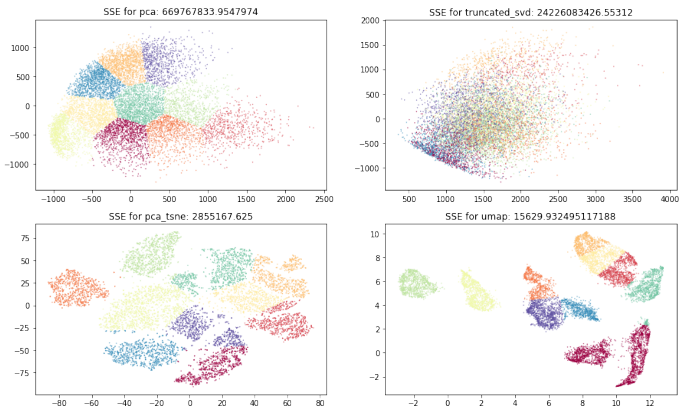


Fig. 6. Clustering using K-Means with some feature reduction technique

V. Results

Using my K-means algorithm with euclidean as the distance metric and best_sse as the initial centroid calculation metric, and using the non-linear feature reduction technique (UMAP) I got a 0.79 V-Score on miner2 for the image dataset. For the Iris dataset, I got a 0.95 V-score on miner2 using PCA and finding correlation as a feature reduction technique. The final submitted cluster for Image data looks like this

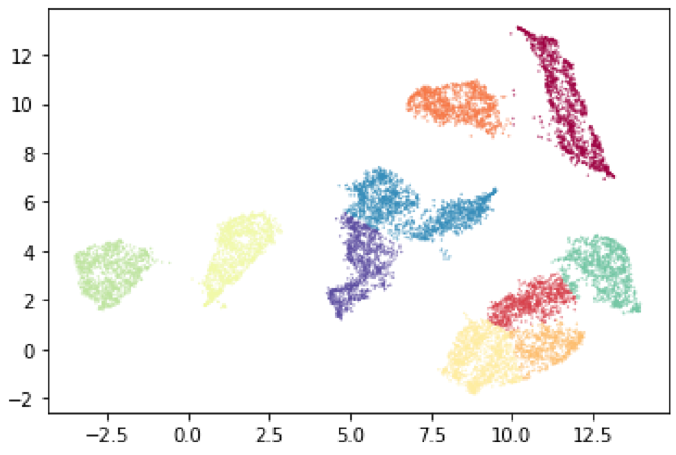


Fig. 7. K=10 Clusters with UMAP and K-Means