Assignment-2

K-Means clustering

Varshith Vootla (Varshith@buffalo.edu)

UBID: 50365537

Abstract

Implementation of k-means algorithm to do multi-class classification on the CIFAR-10 dataset. CIFAR-10 dataset consists of images of size 32 x 32 with 3 channels which makes the image size as 32 x 32 x 3. To make 10 clusters we assume initial 10 points from the given data randomly as centroids and then make iterations to find the distance between all the data set features and find the minimum points.

Dataset

CIFAR-10 dataset has a total of 60,000 samples of which 50,000 are in the training set and 10,000 are in the test set and each image is of 1024 pixels. Each pixel values lies between 0-255. We are using only the test data to do clustering.

Preprocessing & Normalization

As the images are RGB we first convert them into gray scale to make it of size 32 x 32 x 1 and then reshape them into 1024 sized array. All the pixel values lie between 0 and 255 and we will not have any outliers so we don’t need to convert to normalize the pixel values.

Implementation

The idea of k-means clustering is to find the initial points from the given data set. Here in our data we take 10 points as we know there are 10 clusters in the dataset. From the initial 10 points we calculate the Euclidean distance for all the 10,000 set of data.

1. Take 10 random points and pick it from the data set which are to be used as initial centroids.
2. Calculate Euclidean distances of all the data points from the initial centroids.
3. Make each data point assign to a cluster based on distances obtained from step 2.

4. Now we take these initial assignments and run the same process as above with 50 iterations to make the clustering optimizable.
5. We will stop iterations when centroid doesn’t change.

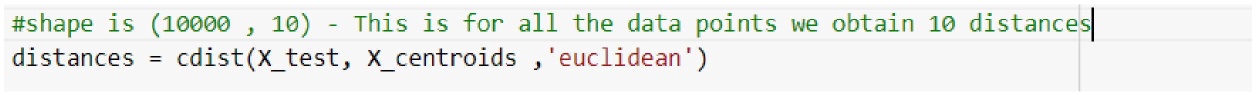
Calculation of distances:

Back implementation of Euclidean distance calculation is

$$d(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

p, q	= two points in Euclidean n-space
q_i, p_i	= Euclidean vectors, starting from the origin of the space (initial point)
n	= n-space

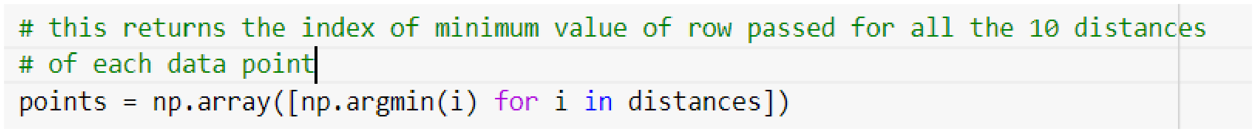
This is implemented from the cdist function imported from ‘scipy.spatial.distance’ library.



```
#shape is (10000 , 10) - This is for all the data points we obtain 10 distances
distances = cdist(X_test, X_centroids ,'euclidean')
```

It returns as shape of (10000, 10). Each data point of test data set is combined with each of the centroids to return a distance. So, we have 10 distances for each data point of test dataset.

Now for each datapoint we need to take the minimum valued distance index (in range 0 to 9). For all the 10000 data points we take these indices which makes the data point belongs to that cluster.



```
# this returns the index of minimum value of row passed for all the 10 distances
# of each data point
points = np.array([np.argmin(i) for i in distances])
```

We obtain distances between the test dataset used and initial assumed centroids

This is to be repeated for 50 iterations with initial assumed clusters and we find the best possible clusters

Model Validation:

Silhouette score and Dunn index are used to validate the data clustering

Silhouette score:

It is a metric with range [-1, 1] with highest value reasons to very good clustering and lowest value -1 leads to bad clustering.

For this assignment we are expecting the silhouette score to be greater than 0.054

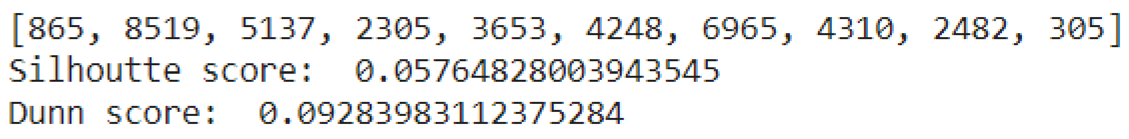
Dunn index:

It is metric to identify the sets of clusters that are compact and having less variance between the clusters

For this assignment we are expecting the Dunn index to be greater than 0.089

Libraries used to implement above indices

```
from validclust import dunn
from sklearn.metrics import silhouette_samples, silhouette_score
```

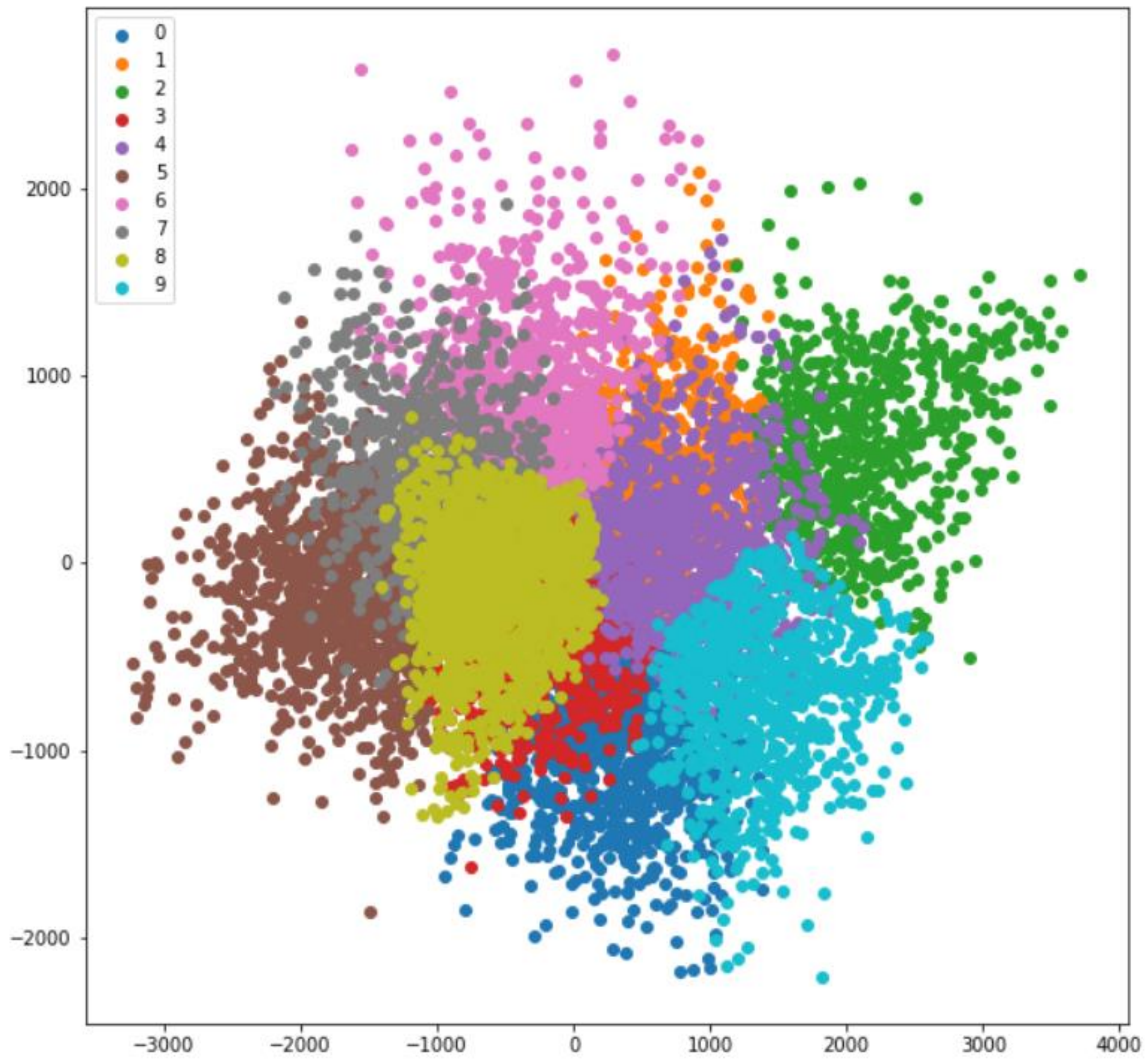


```
[865, 8519, 5137, 2305, 3653, 4248, 6965, 4310, 2482, 305]
Silhoutte score:  0.05764828003943545
Dunn score:  0.09283983112375284
```

For these particular initial random points, we are getting

Silhouette score: 0.057

Dunn score: 0.092



Conclusion:

For the cifar dataset test data after k means clustering implementation we got the Silhoutte score – 0.057 and Dunn index – 0.092

References:

1. https://numpy.org/doc/stable/reference/generated/numpy.argmin.html
2. https://www.google.com/search?q=euclidean+distance+formula&rlz=1C1CHZN_enUS968US968&oq=euclidean+di&aqs=chrome.1.69i57j69i59j0i433i512l2j0i512j0i20i263i512j69i60l2.4708j0j7&sourceid=chrome&ie=UTF-8
3. https://en.wikipedia.org/wiki/Dunn_index
4. https://matplotlib.org/stable/gallery/lines_bars_and_markers/scatter_with_legend.html