Introduction

This report explains and examines two different clustering approaches. Kmeans and Spectral clustering are two different clustering methods that are used to classify data unsupervised. Within different evaluation methods, this report shows how these different clustering perform on different data sets. The rest is organized as follows: first each model and its implementation is explained in one paragraph. Then evaluation and other functions of the clustering.py file is described. Finally, the results section discussed the quality of clustering methods and parameter selection effects. A readme file (Readme.MD) file is provided to show how a user can run the project.

Kmeans Implementation

KMeans clustering is an iterative approach. Kmeans initiate cluster centers with random seeds and assign data points to those random centers based on a distance measure (e.g. Euclidean distance). Then Kmeans updates centers with mean of assigned data points in each cluster and repeats this procedure until there is no more change in cluster centers. Function Kmeans in the python file is written to do the computation I just explained. It gets three arguments, data for data points, K for number of clusters, and eps for minimum amount of changes that keeps algorithm working. Choosing a small number of epsilon will keep the Kmeans algorithm works longer. By setting eps=0.0 the algorithm may never finish on some data sets. Finally the function returns cluster labels for each data point and computed centers.

Spectral Implementation

Spectral Clustering is a bit different than Kmeans. However, it uses K-means as part of its algorithm. I used **KNN** with a user changeable K to construct adjacency matrix. Then I used Gaussian Radial Basis function to form the weight matrix. After computing degree matrix and Laplacian matrix, eigenvectors and eigenvalues were extracted. Later on Kmeans is used to find data points clusters. Before running Kmeans, setting K is another issue we should take care of. While some methods used a pre-defined and fixed K, some others used eigenvalues and the maximum difference gap to find suitable K. I used eigenvalues and maximum gap to find K. However, in some cases K gets a high value. This shows both K as in the KNN can affect K in the KMeans and Spectral Clustering final results. Also, setting K in Kmeans still a big challenge/issue.

Evaluation and other functions

Two types of clustering evaluations are common, one with respect to actual labels which is called external index and the other is based on centers, and clusters' data distribution (internal index). Entropy, Purity, and Normalized Mutual Information were implemented as external index evaluation metrics (function compute_external). Higher values of Purity and lower values of Entropy shows clusters are representing separated classes. For example, Purity of 1 means, each cluster only consists of one class data points. Similarly higher NMI means higher relationship between classes and clusters.

Unlike external index metrics, Internal metrics are evaluation methods to compute how clusters are distributed equally. Internal metrics are more important when there is no actual label. However, with two similar external index values of clustering methods, the one that has better internal index is preferable. I implemented BIC index, Calinski Harasbaz index, Davies-Boulding index, and silhouette index. All metrics were implemented using Rendon et al descriptions [1]. Silhouette index is 1 when data points are distributed densely around their centers and far from other centers (this means a good clustering). Lower values than 1 to -1 means clustering method shows this clustering method is how far from perfect.
The other function in the Python code is visualize that gets data, cluster labels and title and plot each cluster's data points with a different color. Compute distance also computes the Euclidean distance of a point to a series of points and it is implemented for repetitive use.

Results (two sections)
In the main part based on user input arguments there are two options to run both clustering methods. One is a single run with argument of 'Single run' which runs Kmeans and clustering with changeable Ks. It visualizes the data clustering before and after clustering and also compares the implementers codes with ScikitLearn Kmeans and Spectral Clustering w.r.t internal and external index. The other option 'compare' runs both clustering methods with different K from 2 to 20 and plot Entropy, Purity and Silhouette index.

A-Single Run
Figure 1 shows an example of the metrics by running 'single run'. Figure 2. Shows raw data distribution, Kmeans and Spectral clusters with different colors for Iyer dataset. By choosing K=5 for Knn to build matrix adjacency, later Eigenvalues decided two clusters for Spectral clustering while one of them is really dominant. On the other hand, Kmeans clustering seems more successful in dividing data into more balanced clusters. However, it merges multiple classes into clusters at some point. K is defined 5 for Kmeans in this example.
By looking at external index such as Purity we can see, Kmeans perform better in this specific example.
We also compare our implementation with ScikitLearn package and it is similar in Kmeans, but a little bit different with internal index for spectral clustering. This means, they used another method of choosing number of clusters which is fixed K.
By comparing Internal index metrics we can see silhouette index is way higher for Kmeans than spectral clustering in this specific example.
This example shows, choosing parameters is crucial to final results. In the next section we will see clustering methods' performance w.r.t different parameters values.

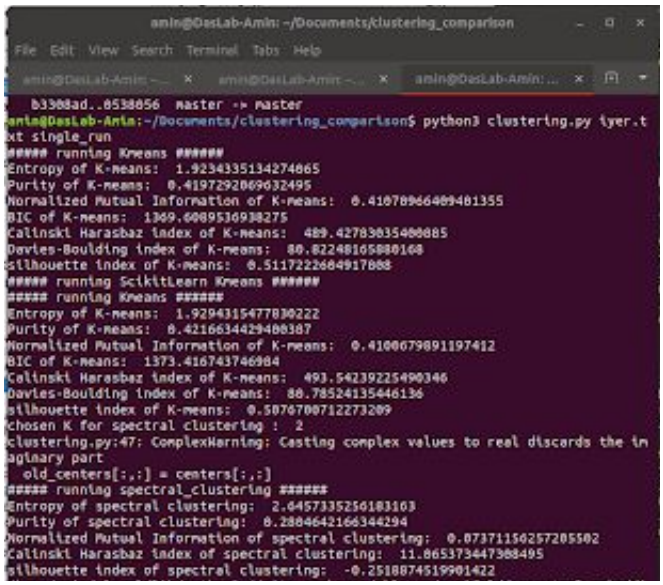

Figure 1. An example of metric. Kmeans with K=5 and Knn with K=5 for Spectral Clustering

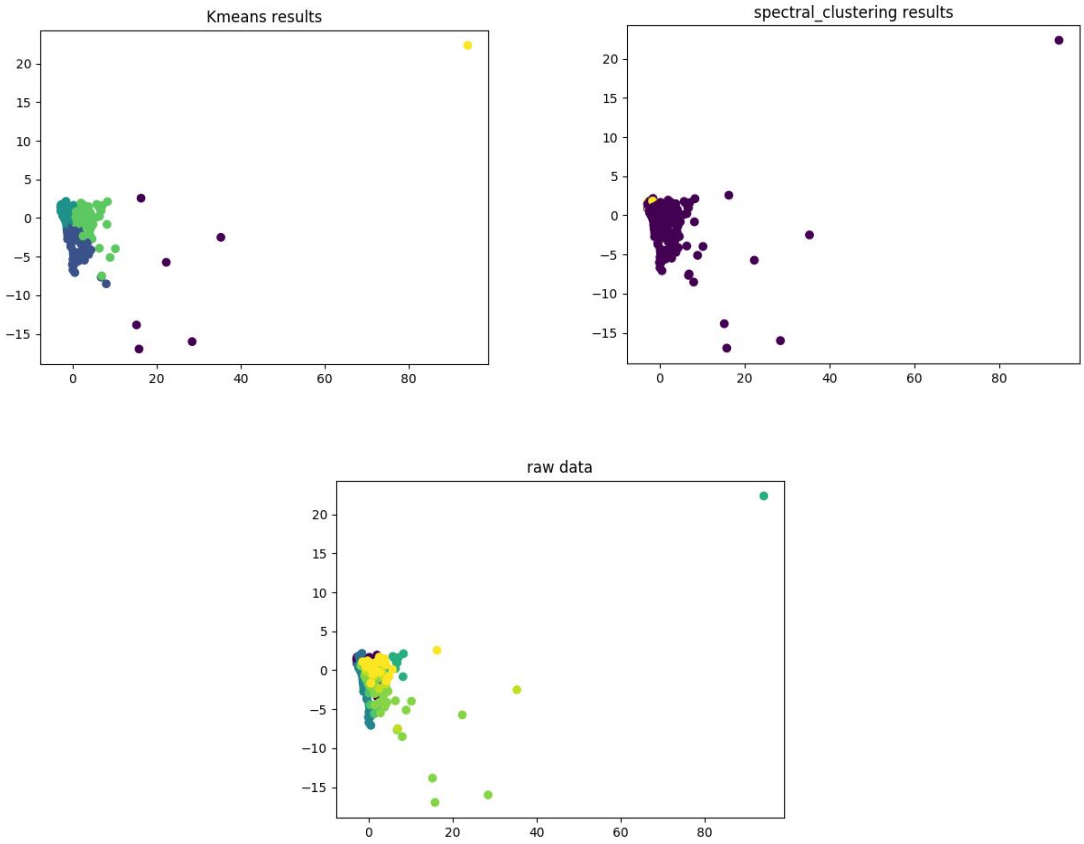

Figure 2. Clustering methods data distribution vs. data distribution with classes as their cluster labels

B-Comparing with different K

In the second set of experiments I used different K's to compare the performance of Kmeans. Figure 2 shows increasing K from 2 to 13 causes steady increment in Purity and steady decrease in Entropy. This is because, by increasing the number of clusters, data points with different classes spread into different clusters. Higher Purity is desired and shows classes are spread less through different clusters and each cluster represents more specific classes.Optimal value is almost 0.7 for K = 14 and 0.72 for K =20. However, if we look at the Silhouette index we will see K=2 has the highest silhouette value and K=14 is among lowest. Higher Silhouette values show, data points are closer to their centers. There is no optimal value for both internal and external index, so maybe a K with high value for both Purity and Silhouette is a better choice. K = 7,8,9, or 10 seems a more reasonable choice.

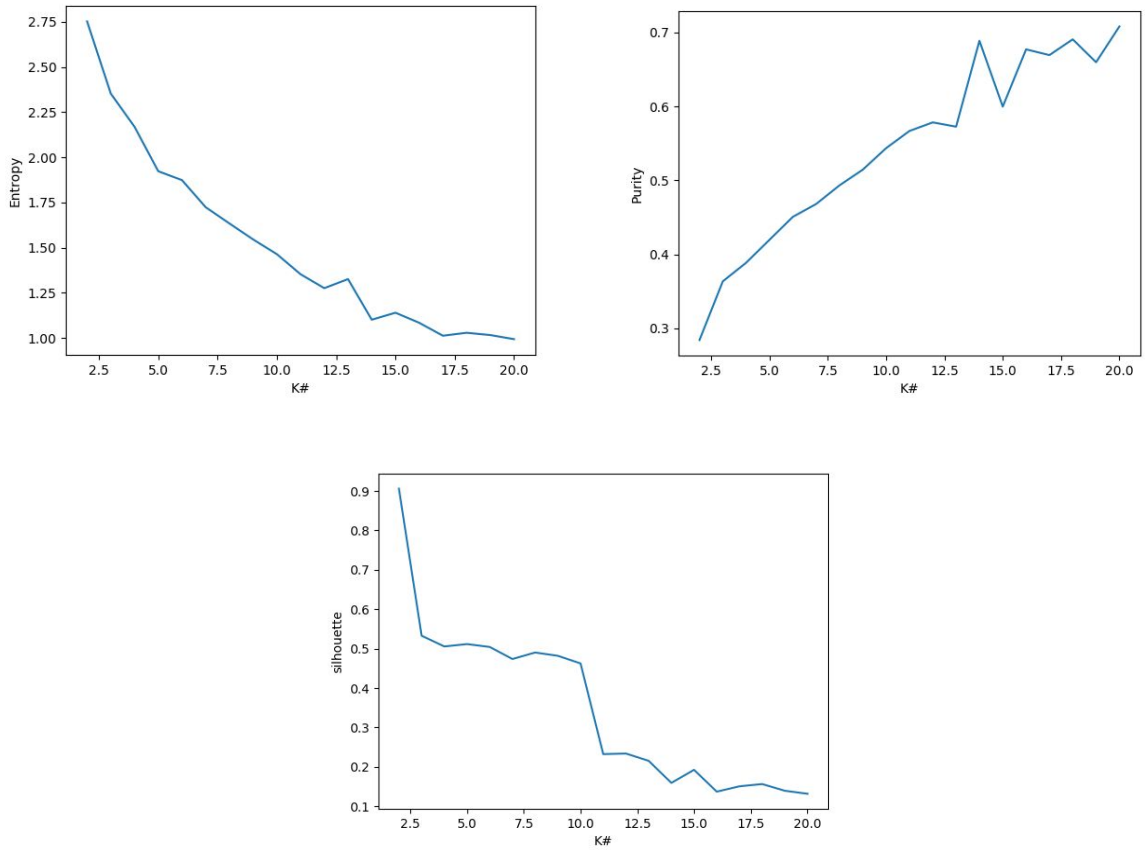

Figure 3. Entropy, Purity, and Silhouette metrics for different K on Kmeans over Iyer dataset

Unlike Kmeans, number of clusters is chosen based on maximum difference in Eigenvalues. Since we used KNN for building adjacency matrix, K for KNN is an effective parameter to our algorithm. Unlike K in Kmeans, changing K in neighbour selection of KNN has a different and non-monotonic relation to final result. Choosing 12,13 or 20 for K, shows highest value for both

internal and external index metrics. However, no run shows Silhouette over 0.0 and that means no matter what value K has, spectral clustering can't divide data points better than Kmean.

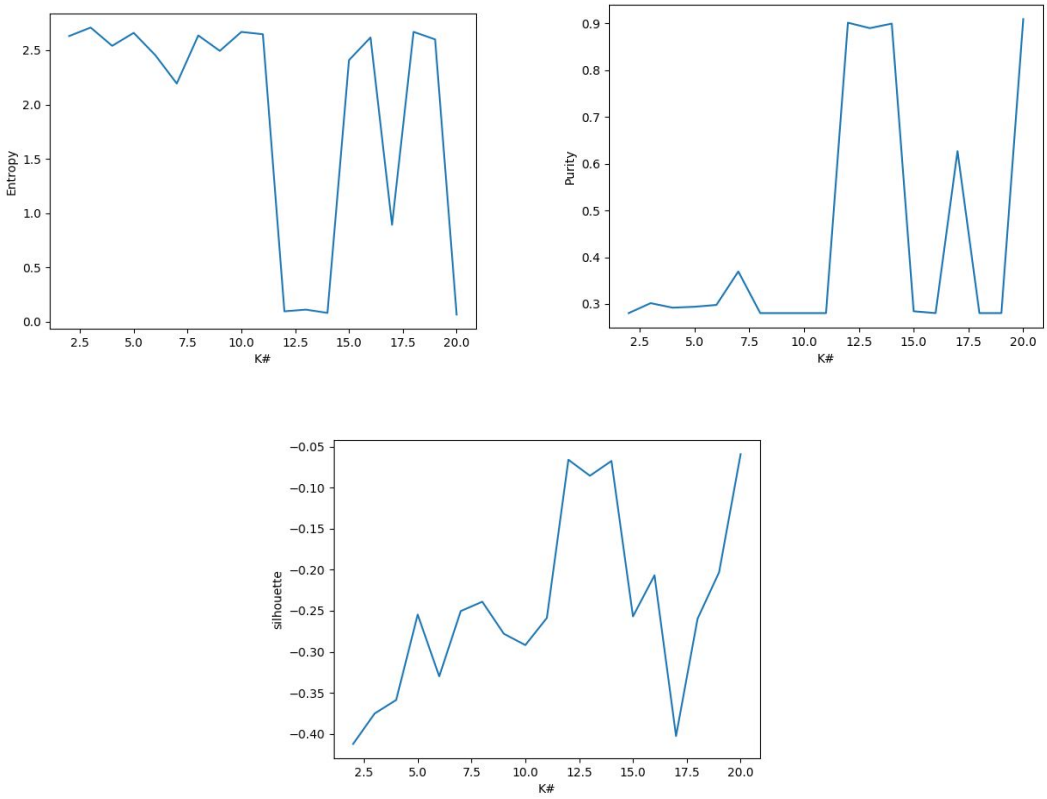

Figure 4. Entropy, Purity, and Silhouette metrics for different K on KNN of the Spectral Clustering over Iyer dataset

Results for Cho dataset are also provided in the pix folder. They show similar results for Kmeans as K goes higher, Purity increases and Silhouette, Entropy decreases. Interestingly, K for K-neighbourhood in spectral clustering has a high value of Entropy for K>= 9 and similarly lowest values. Seems by choosing K=9 we have high support of metrics and the number of clusters are also 3.

References

[1] Rendón, E., Abundez, I., Arizmendi, A., & Quiroz, E. M. (2011). Internal versus external cluster validation indexes. *International Journal of computers and communications*, *5*(1), 27-34.