Homework 5: Introduction to Data Mining (CSE 5243)

Ashwini Joshi
11/6/2017

Introduction:

The report summarizes design choices and results for k-means clustering run on two different datasets. First two sections talk about implementation of clustering from scratch using standard functions in python. In third section, cluster analysis is done using off-the-shelf clustering method in order to compare the results with the implemented algorithm.

Section 1: Description of Code and Design Choices

This section talks about data transformation and implementation of the code. Since all variables are continuous, they are normalized to fall in range (0, 1) using scikit package 'MinMaxScaler' which uses the formula:

New Value = Old Value - min / (max - min)

Where min and max denote minimum and maximum value of the attribute.

The code implements standard Euclidean Distance to find the distance between points of the data. There are no categorical attributes hence it simply sums up squares of the distances along each dimension and takes its square root.

Initial centroids are points chosen randomly from the dataset. Value of k is configurable hence depending upon it, k initial centroids are chosen. Previous set of values of centroids is also stored in order to compute error by calculating distance between current and previous points. Algorithm is iterated over set of such centroid points till all centroids become stable. It also implies if the method is iterated again, points do not change their cluster since centroid for each is stabilized. During each iteration, centroid of a cluster is computed by calculating mean of all points belonging to that particular cluster.

Both true cluster SSE (Sum of Squared Errors) and SSB (Sum of Squares Between clusters) are calculated along with predicted ones. For two dimensional dataset, k is taken as 4 since there are 4 clusters. For wine dataset, k is taken as 6 because quality has values ranging from 3 to 9 for computing true cluster SSEs and SSB.

Output is written to a CSV file which contains ID of the record and its predicted cluster.

Section 2: Analysis of Results

Two Dimensional Dataset

For this dataset, true cluster SSE and SSB are calculated using points that actually belong to a particular cluster to measure the performance of algorithm. The implemented algorithm is run using two values of k (k=4 and k=3).

Table 2.1 is a cross tabulation matrix similar to confusion matrix showing count of actual and predicted number of points for the four clusters. Value in cell (m, n) shows number of points that actually belong to cluster m but predicted as cluster n where m and n denote row and column respectively. For example, entry in the first cell shows the number of records which are classified as cluster 1 and are actually belong to cluster 1. It is observed that most of them are diagonal entries which means that they are classified correctly to their actual clusters.

k=4	Predicted Clusters				
Actual Clusters	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Total
Cluster 1	89	0	0	0	**89**
Cluster 2	2	98	0	0	**100**
Cluster 3	4	1	89	3	**97**
Cluster 4	0	8	2	104	**114**
Total	**95**	**107**	**91**	**107**	**400**

Table 2.1: Cross Tabulation Matrix with k=4 for Two Dimensional Dataset

Figure 2.1 shows scatter plots for both true and predicted clusters. Black dot points indicate the centroid for each cluster. Table 2.2 and Table 2.3 show the comparison between true and predicted values of SSE and SSB with value k = 4. From the tables and scattered plots, it is seen that predicted results are very close to actual results. There is a very slight difference between predicted and true values of SSE and SSB.

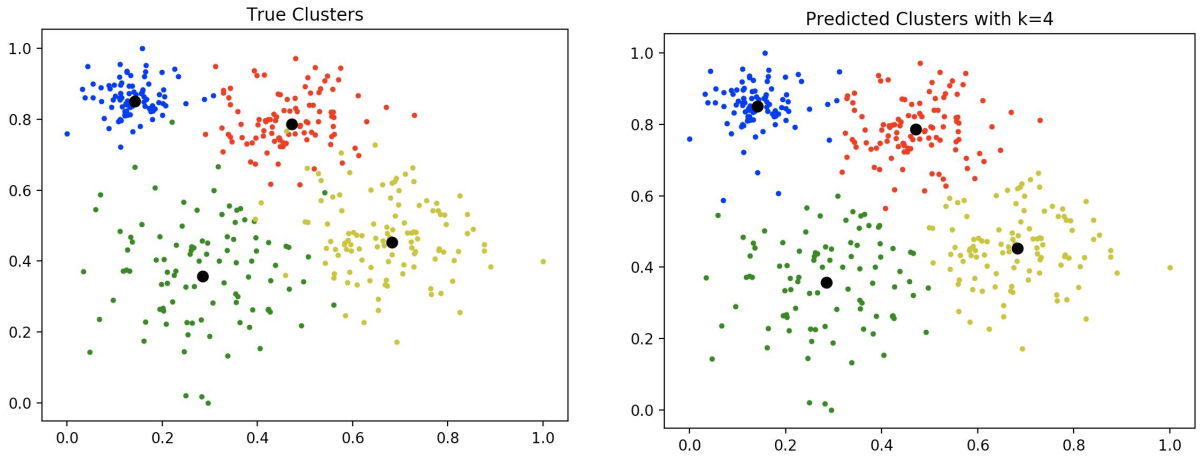

Figure 2.1: Scatter Plots for True and Predicted Clusters

K = 4	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Total
True SSE	0.4354	3.3505	2.6514	1.2587	7.6962
Predicted SSE	0.6917	2.5034	2.0885	1.4966	6.7803

Table 2.2: True and Predicted Cluster SSE for Two Dimensional Dataset

True SSB	Predicted SSB
32.9717	33.8889

Table 2.3: True and Predicted Cluster SSB for Two Dimensional Dataset

Same analysis is done by running the code with value k=3. Table 2.4 is cross-tabulation matrix with three columns as predicted clusters and 4 rows with actual clusters while Table 2.5 shows cluster-wise SSE values along with overall cluster SSE.

k=3	Predicted Clusters			
Actual Clusters	Cluster 1	Cluster 2	Cluster 3	Total
Cluster 1	89	0	0	**89**
Cluster 2	1	110	3	**114**
Cluster 3	5	3	89	**97**
Cluster 4	91	9	0	**100**
Total	**186**	**122**	**92**	**400**

Table 2.4: Cross Tabulation Matrix for k=3 for Two Dimensional Dataset

K = 3	Cluster 1	Cluster 2	Cluster 3	Total
Predicted SSE	3.0437	6.2559	2.7072	12.0069

Table 2.5: True and Predicted Cluster SSE for Two Dimensional Dataset

To find better k-value for given dataset, results are compared using scatter plots and error values. Figure 2.2 shows comparison of scatter plots for predicted clusters with k=3 and k=4.

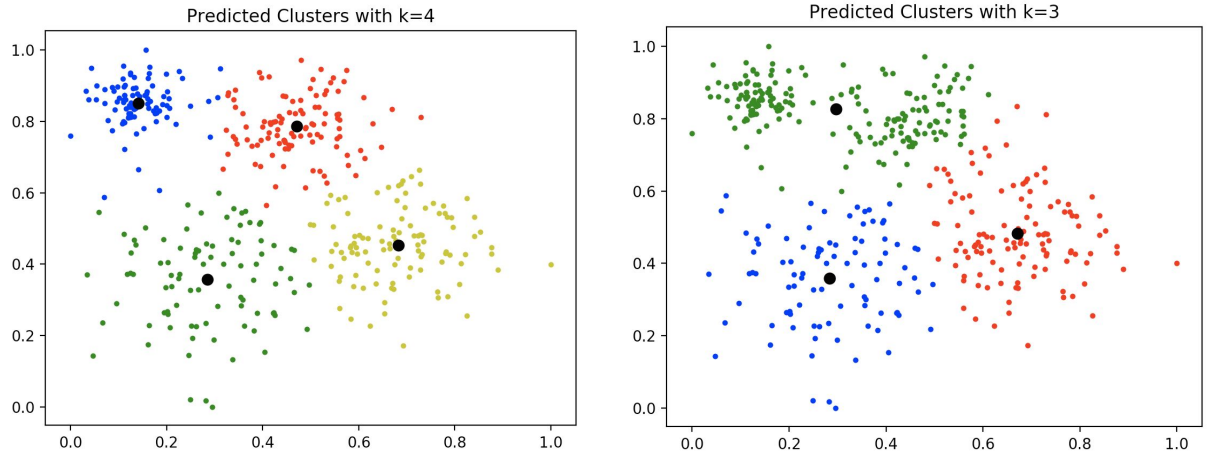

Figure 2.2: Scatter Plots for True and Predicted Clusters

Table 2.6 shows comparison of predicted SSE and SSB values for k=3 and k=4.

Predicted Values	SSE	SSB
k=3	12.0069	28.6620
k=4	6.7803	33.8889

Table 2.6: Comparison of predicted SSE and SSB for Two Dimensional Dataset

From the comparison of the plots in the figure, it is seen that entries in one cluster are merged with another when value of k becomes 3. They are clearly separable into two clusters. In the cross-tabulation matrix as well (Table 2.4), entries in the fourth cluster are merged with the first one. Due to that, distance of the points from its centroid increases which results in large SSE and low SSB in shown in Table 2.6. **Therefore, from scatter plots and error values, conclusion can be drawn that value k=4 is more suitable for the given dataset.**

Wine Dataset

For wine dataset, cluster analysis is done using a three different values of k. Calculated sum of errors are shown in the following tables. Table 2.7 to 2.9 show cluster-wise and overall predicted SSE values for different values of k:

k = 2	Cluster 1	Cluster 2	Total
SSE	122.4262	116.7593	239.1856

Table 2.7: SSE values for wine dataset with k =2

k = 4	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Total
SSE	38.3104	53.1500	43.3018	49.1122	183.8745

Table 2.8: SSE values for wine dataset with k =4

k = 6	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Total
SSE	39.6017	17.9288	25.7831	32.4329	18.7402	24.5541	159.0410

Table 2.9: SSE values for wine dataset with k =6

Table 2.10 shows comparison of SSE and SSB values for values of k mentioned above. It can be seen that SSB values increase gradually as the value of k increases. Also, SSE values decrease with increasing value of k. For k=6, SSE has the least value while SSB has maximum value. From this, it is concluded that k=6 is the best value for clustering of given dataset.

Predicted Values	SSE	SSB
k=2	239.1856	78.2019
k=4	183.8745	123.1255
k=6	159.0410	158.5912

Table 2.10: Comparison of SSE and SSB values for different values of k

Since attribute quality has values ranging from 3 to 8, it can be used to cross validate the results obtained from implementation. There are six distinct values for quality so, k=6 is taken. Table 2.11 shows comparison between actual and predicted SSE.

SSE	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Total
Predicted	39.6017	17.9288	25.7831	32.4329	18.7402	24.5541	159.0410
Actual	2.1456	11.2185	117.5771	120.0400	38.0775	3.7662	292.8252

Table 2.11: Predicted and Actual SSE values for wine dataset with k =6

True cluster SSB is **24.5480** while predicted cluster SSB is **156.9690**. Predicted values for both SSE and SSB are much better as compared to original values since SSE is very high for original and SSB is very low.

Section 3: Off-the-shelf k-means Clustering

To evaluate the performance of implemented algorithm, in-build package from scikit - 'sklearn.cluster.kmeans' is used. The code is written in python where number of clusters is specified. Also, data read from csv file is fitted into required shape before passing it to the package. Normalisation is not necessary in this case because in-built package handles this automatically. After running the package on the data, SSE and SSB are computed for both the datasets and same analysis is done as above.

Two Dimensional Dataset

Table 3.1 and 3.2 show cross-tabulation matrix obtained from scikit package. They are similar to the ones obtained from the implementation with slightly different values.

k=4	Predicted Clusters				
Actual Clusters	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Total
Cluster 1	89	0	0	0	**89**
Cluster 2	0	104	2	8	**100**
Cluster 3	4	3	88	2	**97**
Cluster 4	2	0	0	98	**114**
Total	**95**	**107**	**91**	**107**	**400**

Table 3.1: Cross Tabulation Matrix for k=4 for Two Dimensional Dataset

k=3	Predicted Clusters			
Actual Clusters	Cluster 1	Cluster 2	Cluster 3	Total
Cluster 1	89	0	0	**89**
Cluster 2	1	110	3	**114**
Cluster 3	5	3	89	**97**

Cluster 4	91	9	0	**100**
Total	**186**	**122**	**92**	**400**

Table 3.2: Cross Tabulation Matrix for k=3 for Two Dimensional Dataset

Table 3.3 and Table 3.4 show comparison of implemented algorithm and in-built package SSE values while Table 3.5 shows SSB values for k=3 and k=4 for two dimensional dataset. For k=3, SSE is higher and SSB is lower in scikit but for k=4, scikit has slightly better performance in terms of SSE and SSB values.

k=3	Implementation from scratch	Scikit
Cluster 1	1.4289	1.9594
Cluster 2	1.1982	4.3937
Cluster 3	0.8317	2.2081
Total	**5.0743**	**8.5613**

Table 3.3: True and Predicted Cluster SSE with k=3 for Two Dimensional Dataset

k=4	Implementation from scratch	Scikit
Cluster 1	0.6917	0.5004
Cluster 2	2.5034	1.8446
Cluster 3	2.0885	1.0764
Cluster 4	1.4966	1.4705
Total	**6.7803**	**4.8921**

Table 3.4: True and Predicted Cluster SSE with k=4 for Two Dimensional Dataset

SSB	Scratch	Scikit
k=3	28.6620	24.4387
k=4	33.8889	35.5812

Table 3.5: Comparison of SSB values for Two Dimensional Dataset

Figure 3.1 and 3.2 show comparison of scatter plots for value k=4 and k=3. It is seen that there is no significant difference in clustering the records.

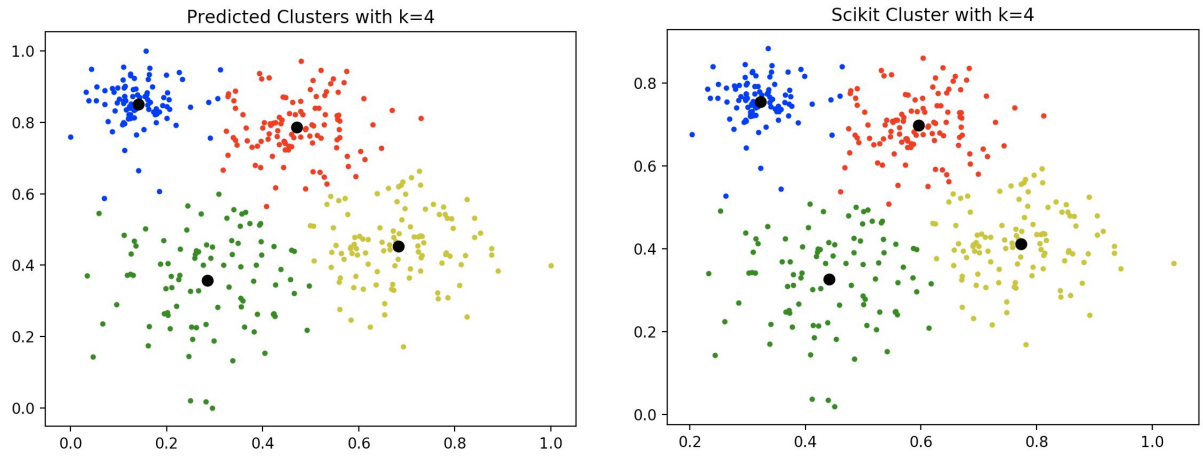

Figure 3.1: Scatter Plots for k=4

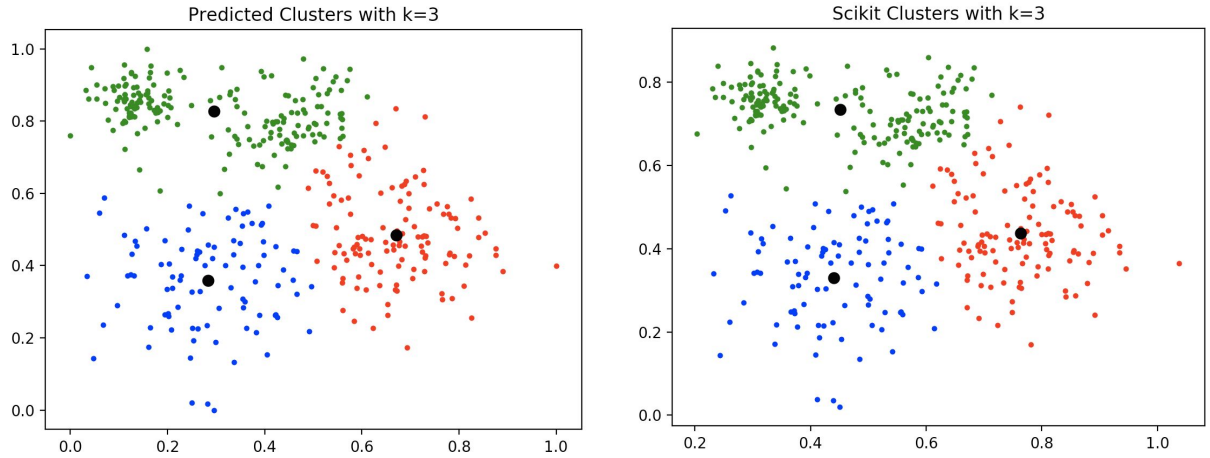

Figure 3.2: Scatter Plots for k=3

Wine Dataset

For wine dataset, analysis is done with three different values of k. These results are compared to that of the predicted values. Table 3.6 to 3.8 show comparison of SSE values with k=2, 4 and 6 for wine dataset.

k=2	Implementation from scratch	Scikit
Cluster 1	122.4262	121.9088
Cluster 2	116.7593	117.2742
Total	**239.1856**	**239.1831**

Table 3.6: True and Predicted Cluster SSE with k=2 for Wine Dataset

k=4	Implementation from scratch	Scikit
Cluster 1	38.3104	49.1122
Cluster 2	53.1500	44.2671
Cluster 3	43.3018	52.6538
Cluster 4	49.1122	37.8351
Total	**183.8745**	**183.8684**

Table 3.7: True and Predicted Cluster SSE with k=4 for Wine Dataset

k=6	Implementation from scratch	Scikit
Cluster 1	39.6017	30.7042
Cluster 2	32.4329	42.3348
Cluster 3	17.9288	20.2320
Cluster 4	24.5541	37.7068
Cluster 5	18.7402	5.1251
Cluster 6	25.7831	20.2755
Total	**159.0410**	**156.3786**

Table 3.8: True and Predicted Cluster SSE with k=6 for Wine Dataset

Table 3.9 shows SSB values for both from-scratch implementation and scikit implementation. From the observations mentioned in the tables above, it can be seen that from-scratch algorithm is giving as good results as the in-built package since all values are in similar range. However, a key point to be noted is that this algorithm is comparatively slower than the in-built one. On datasets containing thousands of records, it might slow down the performance significantly.

SSB	Implementation from scratch	Scikit
k=2	78.2019	78.1902
k=4	123.1255	133.5049
k=6	158.5912	160.9954

Table 3.9: Comparison of SSB values for Wine Dataset

Overall, for both Two Dimensional and Wine dataset, implementation of k-means clustering gives good results. For Two Dimensional dataset, k=4 proves to be the optimal value of k while for Wine dataset, value is 6.