K Means Clustering

Yashwanth Kumar Pamidimukkala

Department of Integrated Systems Engineering, The Ohio State University, Columbus, 43201, pamidimukkala.3@osu.edu

Introduction

[1] k-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster.

Part 1

A.

I built an object-oriented model to implement the K – means clustering model. In this model, the parameters needed are K, tolerance which limits the movement of centroid, maximum iteration which implies now of iterations need to find the optimal centroid point. These three are the three main parameters which need to be passed and the default values are set as well to these parameters. There two functions fit and predict and a constructor.

In the algorithm provided in Jupyter notebook, first we set the centroids using the data points from our data set depending on the value of k. For each record, we will calculate the distances to the k available centroids. We will then classify this record to the cluster whose distance from the clusters centroid is minimum. Later, we will shift the centroid of the cluster and form a new centroid based on the new classification of points made by the algorithm. This function will iterate until the movement of centroid is less than or equal to 0.01. Once the algorithm is developed, I will fit the data into this algorithm and I will calculate all my validity measures such as SSE and SSB. For dataset 1, I calculated SSE, SSB and the tabular matrix.

For the dataset 1:

B.

Below in the table 1 is our computed true cluster SSE and true cluster SSB.
Table 1: True Clusters SSE and SSB

Cluster	1	2	3	4
True Cluster SSE	0.3128	0.9025	2.4301	1.9107
True Cluster SSB	0.2914	0.1622	0.2343	0.2705

C.

Now, we will fit our dataset 1 into the algorithm with k = 4. We compute the SSE and SSB of the clusters formed by our K means algorithm

Table 2: K means cluster SSE and SSB for K = 4

Cluster	1	2	3	4	Total
K Means Cluster SSE	0.5004	1.0764	1.4705	1.8446	4.8919
K Means Cluster SSB	0.1873	0.0558	0.1413	0.1169	0.5013

1. Below are the scatterplots for clusters from dataset 1 and clusters formed by K means Algorithm

Fig 1: Scatter Plot Representing Clusters from the data set 1

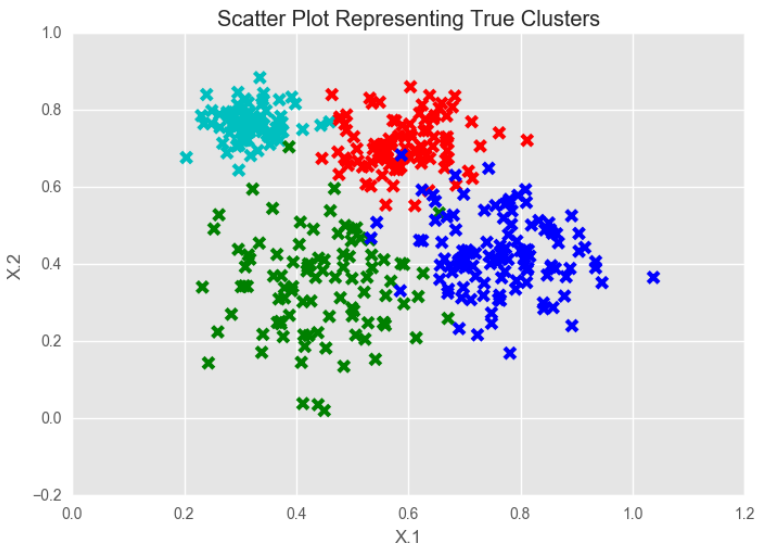
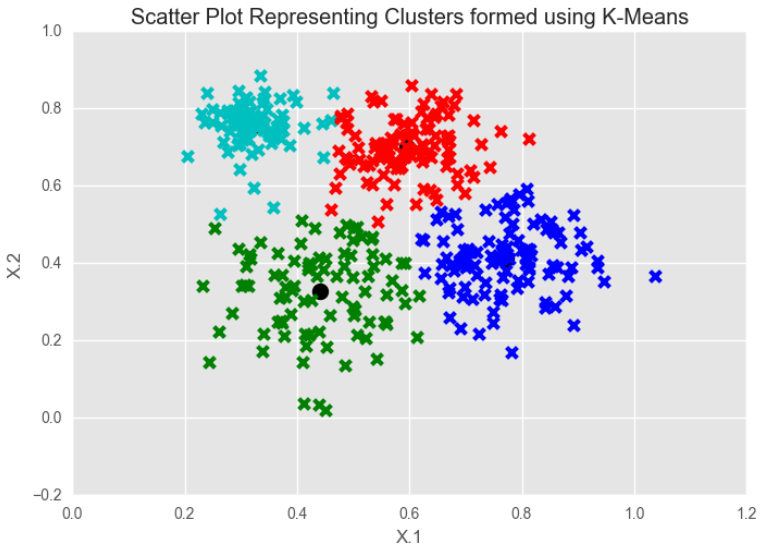


Fig 2: Scatter Plot representing clusters formed by K Means Algorithm



2. Below is table 3 which shows the cross-tabulation matrix comparing the actual and assigned clusters

Table 3 Cross Tabulation Matrix for K = 4

True Cluster K Means Cluster	1	2	3	4
1	89	6	0	0
2	0	94	13	0
3	0	0	84	6
4	0	0	0	108

D.

We will now change the number of clusters to 3 i.e. k = 3 and compute SSE and SSB for each cluster.

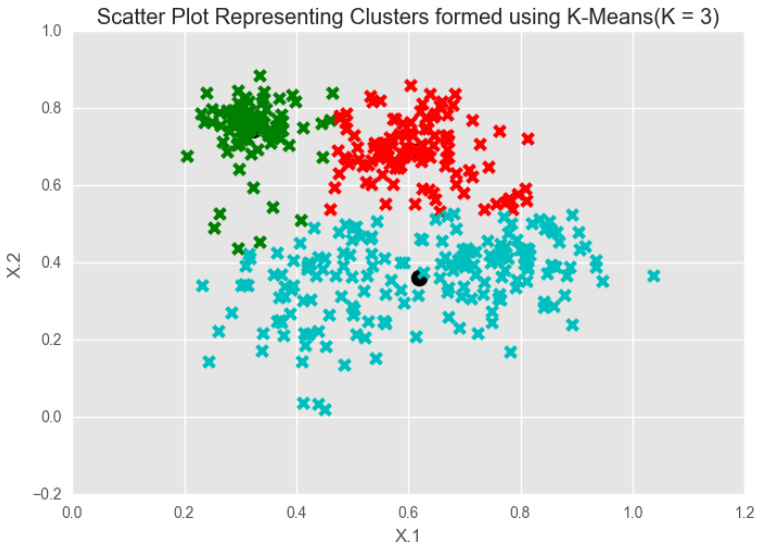
Table 4: K means cluster's SSE and SSB for K = 3

Cluster	1	2	3	Total
K Means Cluster SSE	0.8206	1.5119	8.0768	10.409
K Means Cluster SSB	0.1862	0.0458	0.0910	0.323

Table 5: Cross Tabulation Matrix for K = 3

True Cluster K Means Cluster	1	2	3	4
1	89	10	0	0
2	0	90	27	0
3	0	0	70	114

Fig 3: Scatter Plot Representing Clusters formed by K Means Algorithm for K = 3



From the above results, we can see that SSE of the clusters and total SSE has significantly increased when we changed the values of k from 4 to 3. This implies the clusters are loosely held. Sum of Squared distance between is reduced between the individual clusters which implies they are not so far away and there is chance to overlap too. If we look at the cross-tabulation matrix for K = 4, we can notice the purity of actual

cluster in the predicted clusters. If the actual cluster was used as an external Index, our results show that the K means performed very well in forming tight and independent clusters.

After changing the value of K to 3, we can see from our cross-tabulation matrix that cluster 1 and cluster 2 have high purity but cluster 3 has very low impurity in fact you cannot properly assign cluster 3 to a label. This tell us that K = 4 is a better choice for performing clustering on our dataset 1. In addition to that, if we look at the scatter plot we can see that bottom portion of the plot is one big cluster.

For the Wine Dataset:

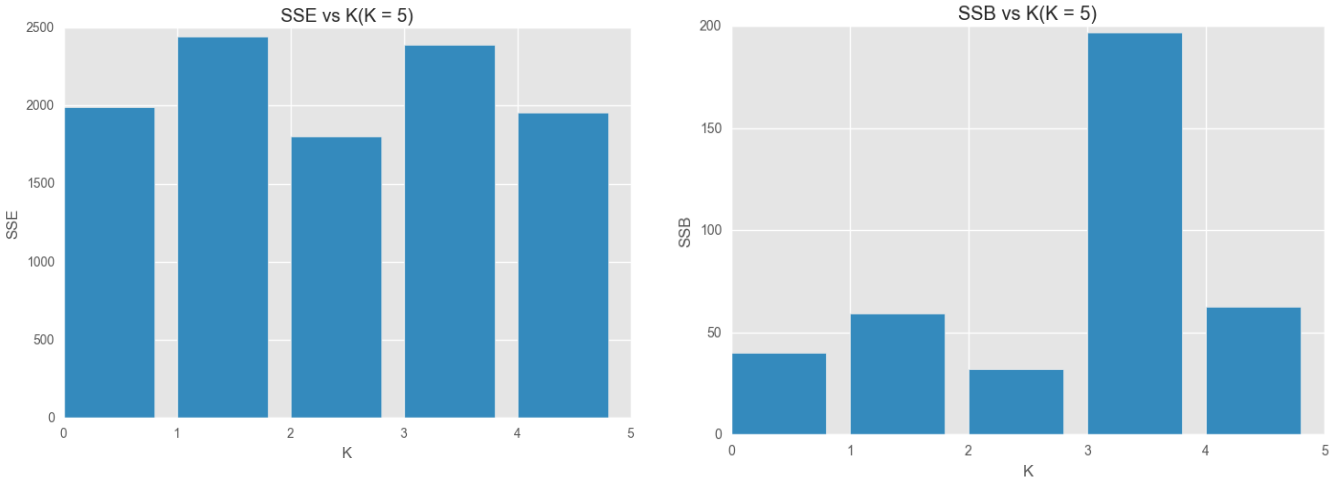
E.

For this step, I initially started with k = 5 as this will help us to perform external index validation on the wine dataset where quality is our externally supplied label for our clusters formed. Later, I increased the value of k to 7 and then to 9. I computed SSE and SSB for each k and plotted graphs for SSE and SSB to show the change in the squared measures for change in the value of k.

F.

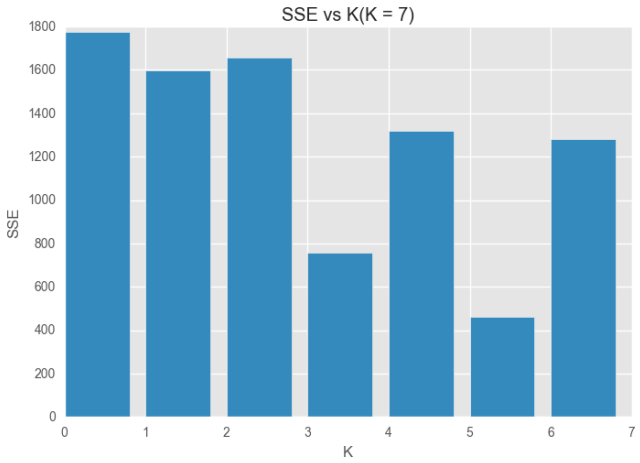
We will look at histograms plotted for SSE against K to show the changes in SSE and SSB for change in value of K

Fig 4: SSE and SSB for K = 5



As we can see from the figures 4,5 and 6 where SSE and SSB is plotted against the values of K, SSE is decreasing for the clusters and SSB is increasing. This shows that as we increase the values of K, our clusters tend to become tighter and tighter and each cluster tends to move far away. For this assignment, I have not iterated over the values of K to find the best value of K, that seems to be a lengthy process. For understanding the effect of change in value of K, I have plotted graphs and compared these graphs with different values of K. Over the values I tested, SSE and SSB for K = 5 are approximately equal to 10585 and 388 respectively and then for K = 7 are 8845 and 1063 respectively and for K = 9 are 7964 and 1262 respectively. These values are only a reflection of what we noticed in our histograms. Over the values I tested I prefer to use K = 9 as it has the lowest SSE and highest SSB.

Fig 5 SSE and SSB for K = 7



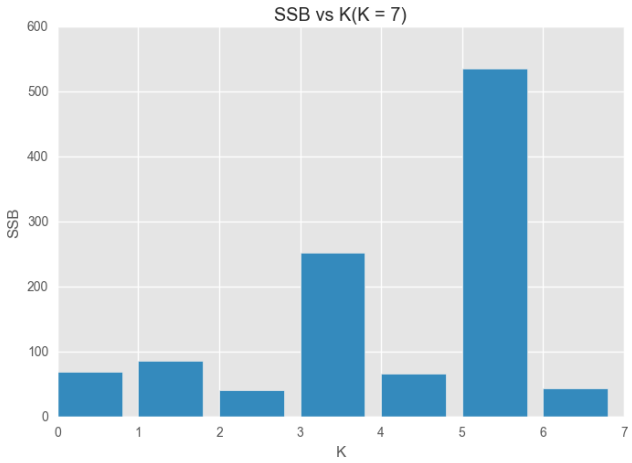
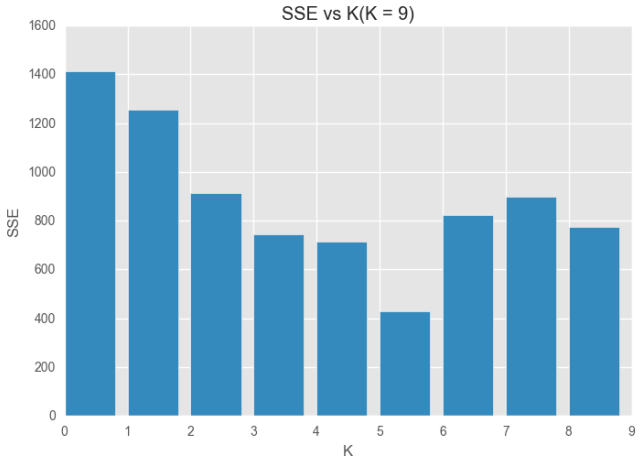
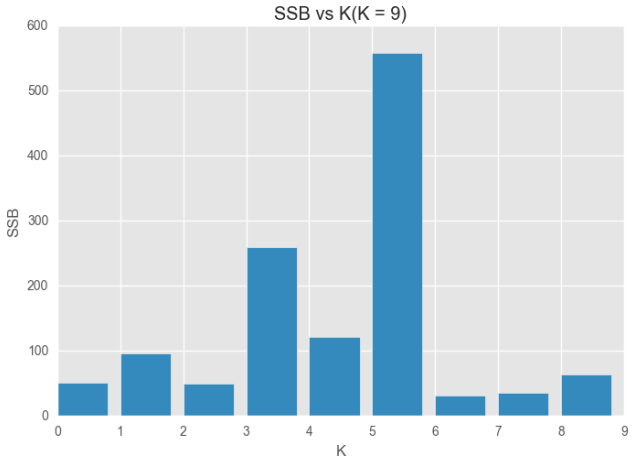


Fig 6: SSE and SSB for K = 9





G.

Table 6: Cross Tabulation Matrix of Quality and cluster K = 6

Quality / Cluster	3	4	5	6	7	8
1	2	16	248	192	61	8
2	2	14	188	131	28	2
3	1	5	80	154	87	4
4	0	6	20	19	1	0
5	5	12	137	123	21	4
6	0	0	8	19	1	0

Table 6 shows the cross-tabulation matrix for cluster and the quality attribute. I chose the value of K to be equal to 6 as we have 6 unique values under quality, this will help us to understand the purity of each cluster. The clusters obtained from our algorithm do not clearly distinguish our data based on quality.

Purity of each cluster seems to be low as we have almost equal number of quality values in a single cluster. This implies the clustering did not do so well in portioning the data based on their quality.

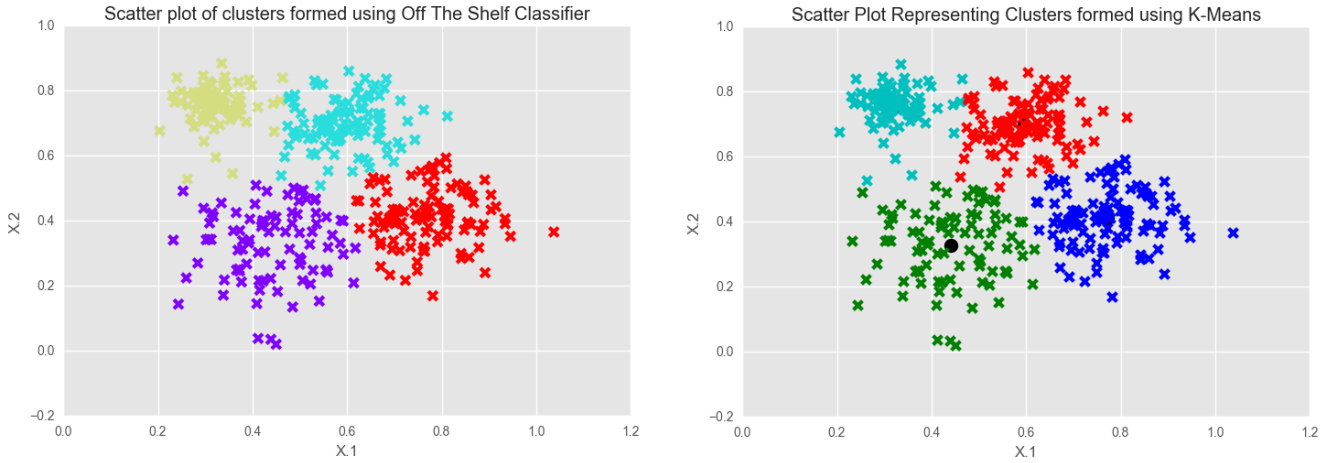
Part 2: Off-The-Shelf Classifier

1. I used [2] sci-kit learn clustering module to implement K Means Clustering. The parameters that I chose were K = 4, max_iter = 300, tol = 0.01 and random_state = 10. All these parameters have the same values which I used in my algorithm except random state. These are the parameters settings used for both the datasets. For wine data set, I used k = 6 for comparing the centroids and the SSE total of the available classifier vs our model.
2. This part has 2nd and 3rd bit from the homework question combined into this one section.

Results:

For Dataset 1:

Fig 7: Scatterplots of K means clusters from off the shelf and model used in this assignment



From this scatterplot, we can see that the results are pretty similar and the off-the-shelf performed the same way as our K means model. Now, we can compare our Sum of Square distances as well using Inertia attribute from sci-kit learn which gives us total SSE. Inertia attribute from sci-kit learn gave a value of 4.89 which is exactly the value of total SSE which we got for dataset 1 (Table 2). We can say that our model is working well. To further confirm our model, I verified the centroids produced by each model and they are the same.

For Dataset 2:

Since we cannot plot scatter plots for this section, the results we would look at is mainly the inertia attribute from the sci-kit learn which gives the total SSE for the wine dataset. Also, we can look at the centroids and compare them with our model. The total SSE of model from sci-kit learn produced a value of 9364.94 which is very closely equal to SSE provided by our model which 9364.08. The centroids that were provided by the off-the-shelf classifier were also very similar to the centroids provided by our model. I have not included the centroids for wine dataset in the report due to the limitation on number of pages.

As we can see in table 7 and 8, the metrics and centroids for both the models are very similar. Also, when the value of K changes, the total SSE decreases as K increases

Table 7: SSE of off the shelf and our K Means Model:

	Dataset 1	Wine
Total SSE(Off-The-Shelf)	4.892	9464.94
Total SSE (Designed Model)	4.891	9364.08

Table 8: Centroids for Dataset 1:

	Centroids (Designed Model)	Centroids (Off The Shelf)
Cluster 1	0.32221294, 0.75482519	0.32221294, 0.75482519
Cluster 2	0.59580368, 0.69767513	0.59580368, 0.69767513
Cluster 3	0.77289734, 0.41198897	0.77289734, 0.41198897
Cluster 4	0.44101165, 0.32613108	0.44101165, 0.32613108

Conclusion:

The model I developed for K Means clustering seems to perform well for the given dataset i.e. Dataset 1 and wine dataset. For K = 4 in dataset 1, the clusters formed by our model are very similar to the true clusters as it is evident from the scatterplot as well. Also, when we look at cross tabulation matrix of true clusters vs clusters provided by our model we can see that purity of each cluster is very high. We calculated SSE and SSB and compared with the values produced by the model when K = 3 and noticed that K = 4 produces better clusters.

We fit the wine data set into our model and varied the value of K and noticed that SSE is decreasing while we were increasing the value of k. From the values I experimented, I chose K = 9 as it has lower SSE and higher SSB than other values of K. Comparing these clusters with quality attribute didn't provide any significant results as the purity of each cluster was very low. Finally, we used off-the-shelf classifier to verify the results from our model and they both seems to be producing the same results.

References:

[1] https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm

[2] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.