Machine Learning
Take Home Exam
Kmeans

Danish Hafeez
Dept. of computer engineering
Air University
Islamabad,Pakistan
171458@students.au.edu.pk

***Abstract*—This document is about my Take Home Exam of machine learning in which I implemented K-means algorithm on the random Gaussian Data .**

I. Introduction

We are given the number of years of experience and number of cases handled of each individual. Based on the two features, we are required to deploy k-means clustering algorithm and come up with the formulation of teams, who can start working on the relief efforts. *K*-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable *K*. The algorithm works iteratively to assign each data point to one of *K* groups based on the features that are provided. Data points are clustered based on feature similarity. The results of the *K*-means clustering algorithm are:

1. The centroids of the *K* clusters, which can be used to label new data
2. Labels for the training data (each data point is assigned to a single cluster)

Rather than defining groups before looking at the data, clustering allows you to find and analyze the groups that have formed organically. The "Choosing K" section below describes how the number of groups can be determined.

II. Dataset Generation

I was asked to create a random data given mean points (3, 70), (7, 150) and (13, 250). Take these values to be the mean of three different Gaussian distributions, generate 100 random data samples for each mean. Generate the data using standard deviation to be 3 in each dimension and then added Gaussian noise in this data our final Data set was of 2x300.

III. K-Mean Algorithm

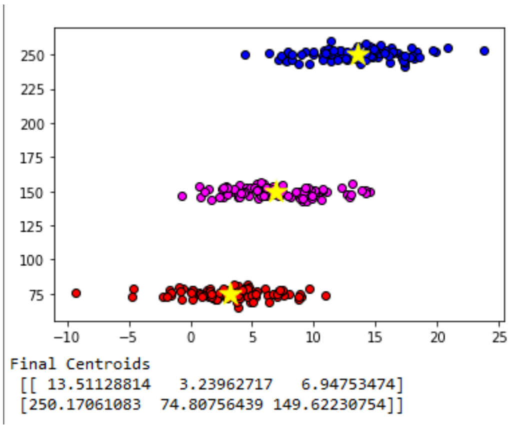
After Generating Data I was to implement K means clustering on the Data set. I had all my data in the variable named ”X_Final” during algorithm implementation I used the data in 300 x 2 format by taking transpose of my dataset this helped me to access the columns of final dataset easily

Than moving to Algorithm, Step 1 was to compute Random Centroids (mean points) as I want to make 3 clusters for this I computed random numbers using np.random.randint to initialize my centroids

Than after initialization I moved to the second step which was to compute which data point is closest to which centroid this was computed by using argmin of np first I calculated the distance of every data point in X_Final than took square of it after that I had a 1 x 300 matrix which tells me which set is nearer to which cluster.

Now it was to update the mean values in our case those were three of them I updated these values using the concept by which I took the mean of clusters belongs to each previous centroid than updated it as my new centroid

The upper two steps were repeated until previous mean and new mean difference is less than E=0.001.Aftter that I had new updated centroids I used matlplot.lib to plot it and observed my code is working okay or not



```
Final Centroids
 [[ 13.51128814   3.23962717   6.94753474]
 [250.17061083  74.80756439 149.62230754]]
```

Figure 1:Final figure

I iterated the code 20 times and observed the result I figured out that the initialization of centroids is an important step If any how two centroids were selected from one initial cluster than one centroid is assigned to it the table for it is Given on the next page we can observe the location of initial means had a significant effect on the cost of our algorithm

Itteration #	Centroids(Final)	Cost
1	[[2.0331218, 73.12085421] [4.19128668 77.50164584] [9.67271257 200.3769717]	[3.1801791091912355, 2.376844382347413, 2798.878903697938]
2	[7.7128238 149.55978556] [12.296099 250.01263993] [2.9861733 75.44874932]	[38.595701282500755, 24.939587556055066, 22.066659243260563]
3	[6.8103555 149.68632108] [13.1467736 250.016459] [2.65039996 75.56058639	[14.915630434029943, 5.606278294881463, 0.3396877427106648]
4	[5.05257592 112.4645503] [11.4547134 247.0346643] [14.5144082 252.8980506]	[2167.4784862620613, 31.695605665777002, 6.704254090530988]
5	[2.41129684 74.95369389] [12.639316 250.3427741] [6.99082528 149.9892648]	[28.94608529059635, 33.534668310664856, 15.178643318059462]
6	[7.23677245 150.2657543] [12.9785602 249.785753] [3.05395823 74.62965139]	[3.6148819729208346, 21.736458660285045, 28.617135149636862]

IV. K-Mean Analysis

A. Which step was computationally most extensive?

According to my analysis and understanding of mathematics behind it the update of new Centroids is most comput-ationally expansive step

B. How many features which were generated from the same Gaussian in step (i) of 'Dataset Generation', got clustered together?

I computed with taking Random initial Centroids Some time they form the Clusters with equal number of Data points Some time two centroids were closer and one was far away this was the worst case in above table 1st and 4th iterations are examples of Worst case

C. Why some features of the same Gaussian did not get clustered together?

As mentioned Earlier I computed Centroids Randomly So it there was a chance that two centroids will clusters in same Gaussian ,also in K- Means clustering there is no inner relation between centroids So this Anomaly is caused

D. Based on the given scenario, what is a reasonable clustering result in terms of the application?

Based on the given scenario, a reasonable clustering result in term of application was Cluster-than-Predict in which we made a model of 3 subgroups

E. Does our algorithm give a reasonable enough result?

Overall our algorithm gave a reasonable result but we could have obtained best if instead of Randomly initializing centroids we could just have some Estimation or through data analysis we could have predicted the groups(like in our case it was visible for Developer that there are three classes)
And chose our initial centroid from that group etc

F. What numbers can we put in the table to show that the performance of our algorithm is correct?

To check the performance of our algorithm we can find a relation between convergence rate and Cost

V. Conclusion

Our algorithm did a satisfying job in clustering but the anomaly of not clustering in same Gaussian because of random initialization of centroids can cause huge problem So modern Estimation or data analysis techniques should be used to obtained best accuracy.