HW 08: DBScan
See the associated Dropbox for due date.
Thomas Kinsman

This assignment is like kMeans. The data is very similar. The main difference is the presence of noise.

Homework is to be programmed only in one of the following languages. No other languages will be accepted. Please limit yourself exclusively to: Java, Python, Matlab, or R. The last three have good native graphics and plotting support.

Assume that the grader has no knowledge of the language or API calls, but can read comments. Use prolific block comments before each section of code, or complicated function call to explain what the code does, and why you are using it. Put your name and date in the comments at the heading of the program. **Worked with Manan Shah.**

Hand in IN A ZIP DIRECTORY with your name on it: HW08_LastName_Firstname

1. Your write-up including your results, **HW08_LastName_FirstName_kMeans.pdf**
2. Your well commented code, **HW08_LastName_FirstName_kMeans…**

You are provided with a file of data. This data has only three attributes to select from.

Implement the DBScan Clustering Algorithm: (Section 8.4 of the textbook.)

1. The data file is provided. (Which is purely synthetic, but good enough to test the algorithm.)

2. The data points represents stars in space. The three attributes are (X,Y,Z) coordinates. The units are "Astronomical Units". Your job is to identify the galaxies in this star field using DBScan clustering.

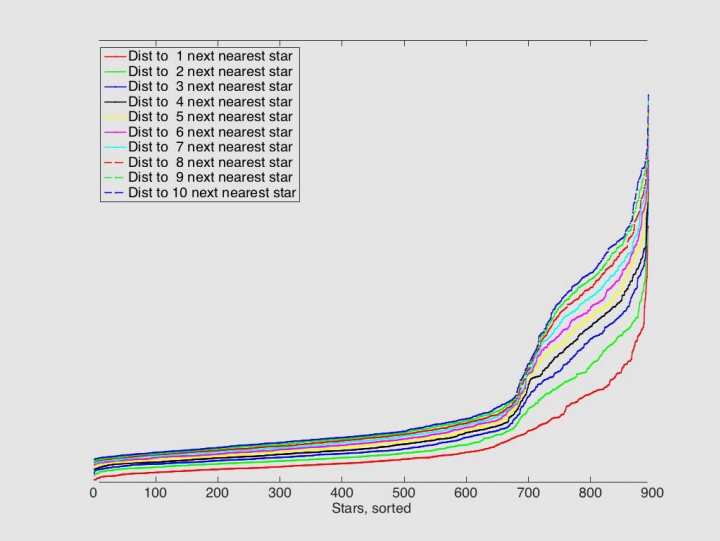


Figure 1 Distances to the 1st nearest neighbors, the 2nd nearest neighbors, the 3rd nearest neighbors, … etc…

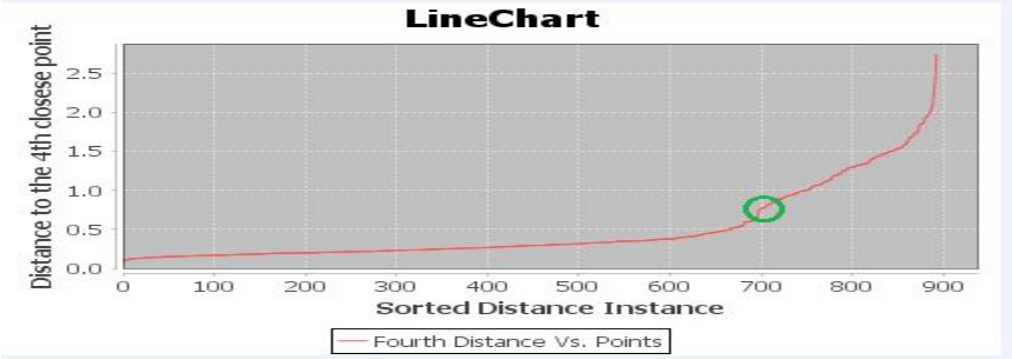
sorted from minimum to maximum for each distance.

(Graph is intentionally missing the Y axis, so as not to spoil the fun.)

3. **Write-UP Should Include:**
   a. **Noise cleaning identification:**
      i. For each star, compute the distance to the 4th closest star.
      ii. Sort these distances and plot them.
      iii. You should get a plot that looks somewhat like the forth line in Figure 1. iv. From this plot, you should be able to find the jump in the distances that occurs from the noise points.
      v. NOW, the issue is to find a noise removal rule.
      Look at the slopes of the graphs, and find a distance from the point that rises the furthest the fastest. You look for a line that rises more than others, and helps best to differentiate the "inside a cluster" points from the "noise" points.
      vi. From this look for the **inflection point** (as described in the textbook) and use the associated distance as the eps distance for noise. Any cluster that has a forth nearest neighbor that is over eps (short for epsilon) is considered a noise point, and ignored. (This is an example of a changepoint detector, which has wide and varying uses.)
      vii. For example, from the above graphs, we might say that when distance to the 4th closest star is over 1.2, the point is probably a noise point.
      From this look for the inflection point (as described in the textbook) and use the associated distance as the eps distance for noise. Any cluster that has a forth nearest neighbor that is over eps (short for epsilon) is considered a noise point, and ignored.
      viii. Add your own graphs and figures to show what you discovered.
      ix. Describe what parameters you decided to use for eps.

   b. What distance metric did you use? USE THE EUCLIDEAN. However, in the real world you would use something that described the distances between people. Especially for social network analysis.

   c. Using DBScan, (see the textbook). Identify the number of clusters.
   5 Clusters. (Eps value=0.75)



d. Sort the clusters from smallest to largest. For each cluster, create a table and report:
   i. How many data points were in it?
   ii. What was the center of mass of this cluster?

Cluster Number	Number Of Data Points	Center Of Mass
1	177	[6.96,0.88,6.08]
2	132	[8.08,6.08,2.04]
3	133	[3.04,8.01,4.01]
4	130	[3,3,7.99]
5	126	[4.91,9.07,8.52]

```
setwd("E:/BDA Homework/HW08-DBScan_NOISY")
getwd()
data<-read.csv("HW_08_DBScan_Data_NOISY_v300.csv")
out<-dbscan(data,0.75)
out
```

e. Also, report an estimate of the number of noise points.

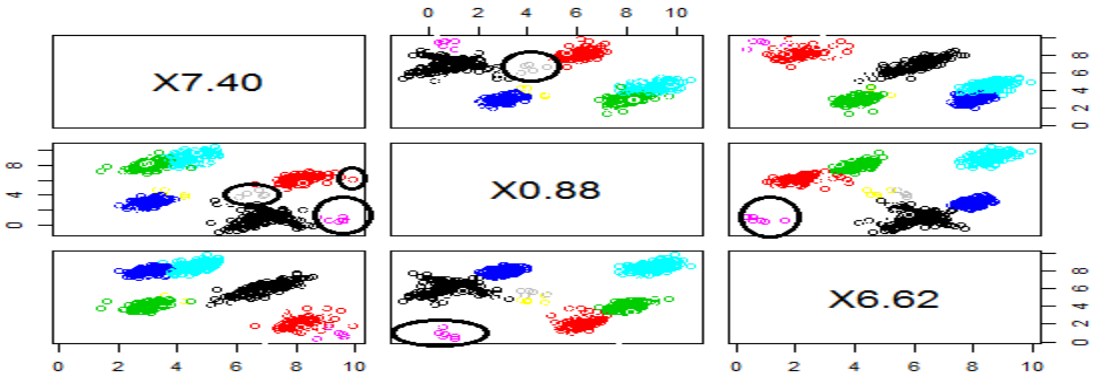
195 Noise Points

f. Write a conclusion:

Describe what you learned from this exercise, how hard it was for you to implement this, compare and contrast this to other algorithms you have implemented, was it harder or longer… etc… Show any graphs or data visualizations you want… give evidence of learning.

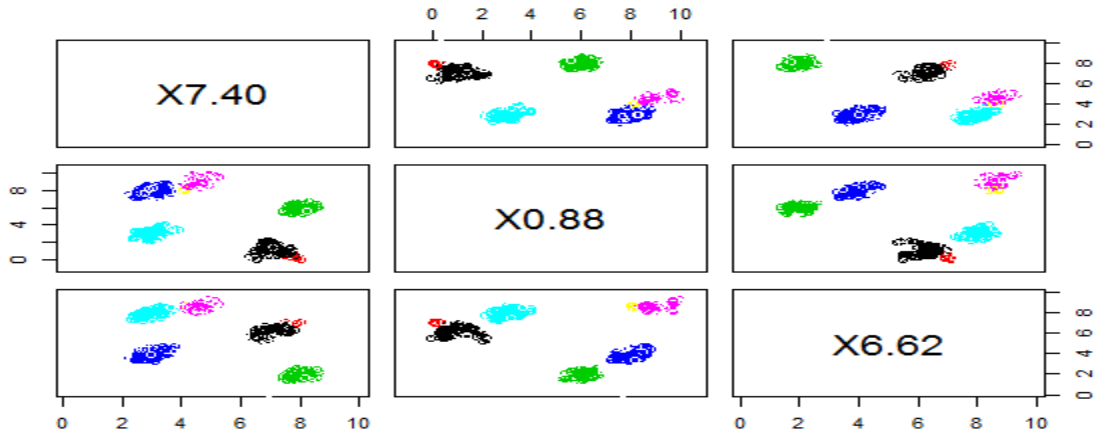
Learnings From this Exercise:

1. DBScan Algorithm works best for clustering the sparse data.
2. This is used where K-Means fails.
3. The most important part about DBScan is choosing the eps value and merging the clusters once the clusters are formed.
4. Checked for different values in R for the different values of eps. Below are the screenshots of the understanding



Above screenshot shows that if we increase the eps (increase the radius of a cluster), we are bound to increase the number of noise points we include in the analysis.

```
setwd("E:/BDA Homework/HW08-DBScan_NOISY")
getwd()
data<-read.csv("HW_08_DBScan_Data_NOISY_v300.csv")
out<-dbscan(data,0.9)
plot(data,col=out$cluster)
```



If we decrease the eps value below than a threshold (0.75), the clusters formed are not very clear and also they are very small in size. This means there is a lot of noise cancellation when we choose the lower eps value.

```
setwd("E:/BDA Homework/HW08-DBScan_NOISY")
getwd()
data<-read.csv("HW_08_DBScan_Data_NOISY_v300.csv")
out<-dbscan(data,0.3)
plot(data,col=out$cluster)
```