CIS6930

Fall 2017

Introduction to Data Mining

Project 2: Clustering

November 7, 2017

Submitted By:

Jaspreet Bajwa

(81811985)

Project Description:

This is an introductory level project, aimed to increase the familiarity with the various clustering techniques/packages available in R to do easy data mining analysis on real-world problems. The main aim of this project was to study the data sets available at the link mentioned in the references and cluster the various data points using four of the clustering techniques mentioned below:

1. Hierarchical Clustering
2. K Means Clustering
3. Density Based Clustering
4. Graph Based Clustering

Data Set:

There are two datasets for performing the cluster analysis. Dataset1 contains 1000 data points of 8 clusters in 3D space. The first 3 columns represent the corresponding coordinates in 3 dimensions and the 4th column gives the ground-truth cluster labels of each point.

	A	B	C	D
1	x	y	z	cluster
2	7.322427	-9.09149	4.434423	1
3	4.468376	-3.42393	5.789986	1
4	4.931896	5.08504	9.92535	1
5	5.779264	3.676984	5.453262	1
6	1.798283	7.400238	6.368287	1
7	-4.17156	-9.73454	0.376833	1
8	5.653729	-1.88853	7.426151	1

	A	B	C	D
1	x	y	z	w
2	540	7.307407	0.528102	0.964815
3	58	5.310345	2.537574	0.258621
4	16	15.5625	1.008097	0.75
5	65	7.923077	1.31	0.584615
6	81	2.641975	3.489286	0.098765
7	88	0.931818	4.937485	0.056818
8	55	5.763636	2.593289	0.218182

The second dataset, named as dataset2.csv, contains more than 1 million data points with the coordinates in 4 dimensions.

Data set preparation:

The purpose of data set preparation is to transform data set so that their information content is best exposed to the mining tools. The aim of this exercise is that the error prediction rate should be lower or the same after the preparation is done.

Normalizing:

There can be instances found in data frame where values for one feature could range between 1-100 and values for another feature could range from 1-10000000. In scenarios like these, owing to mere greater numeric range, the impact on response variables by the feature having greater numeric range could be more than the one having less numeric range, and this could, in turn, impact prediction accuracy. The objective is to improve predictive accuracy and not allow a feature to impact the prediction due to large numeric value range. Thus, we may need to normalize or scale values under unique features such that they fall under common range. I implemented the min-max normalization technique to the data.

$$(X - \min(X))/(\max(X) - \min(X))$$

Clustering Analysis on dataset1.csv:

Hierarchical Clustering:

Hierarchical clustering is an alternative approach to k-means clustering for identifying groups in the dataset. It does not require us to pre-specify the number of clusters to be generated as is required by the k-means approach. Furthermore, hierarchical clustering has an added advantage over K-means clustering in that it results in an attractive tree-based representation of the observations, called a dendrogram.

Parameter selection:

Hierarchical clustering is of two types: Agglomerative and Divisive hierarchical clustering. So, I applied both the method packages: hclust and agnes for agglomerative clustering and diana for divisive. On analyzing the results, I found hclust to be more efficient, so I have used that.

Out of the different methods (ward, single, complete, average, median etc.) I used the 'complete' method.

Clustering distance measure:

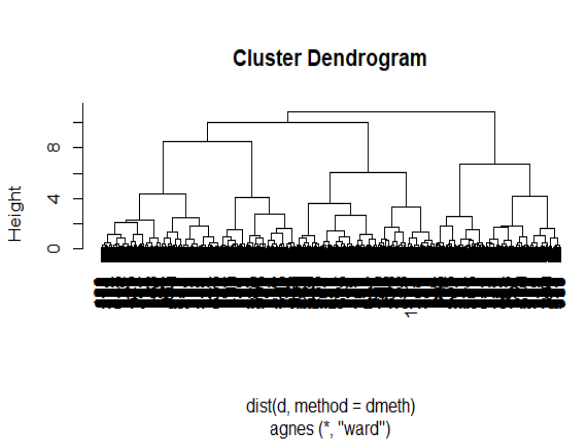
The classification of observations into groups requires some methods for computing the distance or the (dis)similarity between each pair of observations. The result of this computation is known as a dissimilarity or distance matrix. There are many methods to calculate this distance information; the choice of distance measures is a critical step in clustering. It defines how the similarity of two elements (x, y) is calculated and it will influence the shape of the clusters. Out of the various methods available for computing distance, I have used "Euclidean" measure for better results.

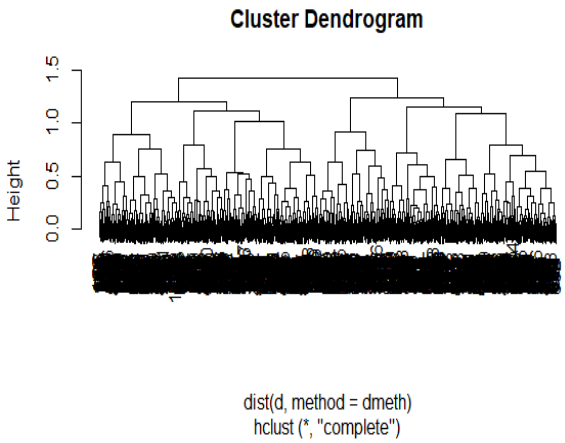
Optimal Clusters (Value of K):

To determine the number of clusters, I used the elbow method, average silhouette and gap statistic method.

Results:

```
Clustering for  KMeansList of 9
 $ cluster     : int [1:1000] 7 3 2 8 6 8 6 2 3 7 ...
 $ centers     : num [1:8, 1:3] 0.4424 0.2034 0.0355 0.1779 0.7569 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:8] "1" "2" "3" "4" ...
  .. ..$ : chr [1:3] "x" "y" "z"
 $ totss       : num 280
 $ withinss    : num [1:8] 6.16 5.73 6.47 6.81 7.41 ...
 $ tot.withinss: num 62.8
 $ betweenss   : num 217
 $ size        : int [1:8] 115 97 99 114 125 122 169 159
 $ iter        : int 4
 $ ifault      : int 0
```





K Means clustering:

K-means clustering is the most commonly used unsupervised machine learning algorithm for partitioning a given data set into a set of k groups (i.e. k clusters), where k represents the number of groups pre-specified by the analyst.

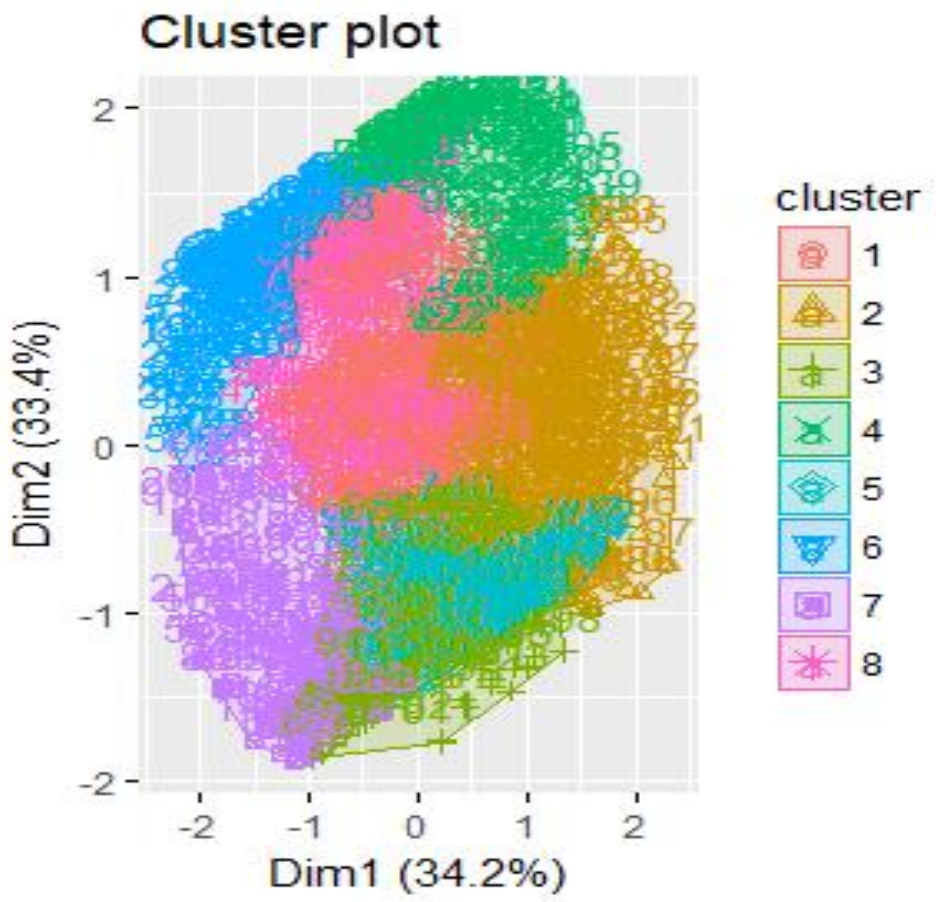
Parameter Selection:

We will group the data into eight clusters (centers = 8) since we need to divide it into eight clusters. The kmeans function also has an nstart option that attempts multiple initial configurations and reports on the best one. For example, adding nstart = 25 will generate 25 initial configurations.

K Means algorithm runs with different algorithms like "Hartigan-Wong", "Lloyd", "Forgy", "MacQueen" etc. Based on the results with each one, I have used Hartigan-Wong due to better efficiency.

Results:

The plot after the clustering method is as shown below: As can be seen, the data is divided into 8 clusters shown by distinct colors.



Density Based Clustering:

DBSCAN stands for Density-Based Spatial Clustering and Application with Noise.

The advantages of DBSCAN are:

1. Unlike K-means, DBSCAN does not require the user to specify the number of clusters to be generated
2. DBSCAN can find any shape of clusters. The cluster doesn't have to be circular.
3. DBSCAN can identify outliers.

To perform density based clustering, I used in built package dbscan.

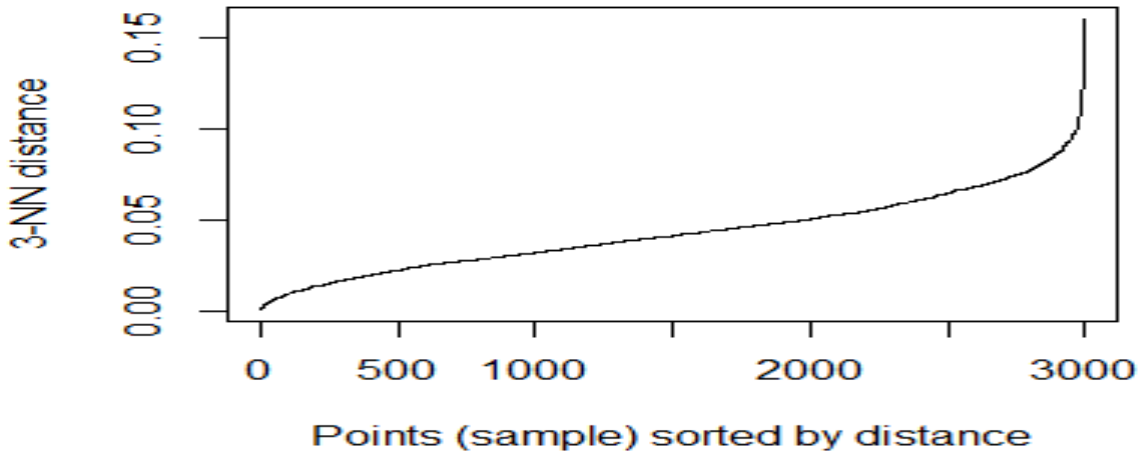
Parameter Selection:

The performance of dbscan depends on the value of eps and minPts.

Calculation of eps:

To effectively calculate the values of eps, I used knndistplot() method with 3 variables for each of x, y and z. The method consists of computing the k-nearest neighbor distances in a matrix of points. The idea is to calculate, the average of the distances of every point to its k nearest neighbors. The value of k will be specified by the user and corresponds to MinPts.

Next, these k-distances are plotted in an ascending order. The aim is to determine the “knee”, which corresponds to the optimal **eps** parameter. At the plot that is generated, we calculate the value of eps by checking the knee in the plot. With the dataset1, eps value came out to be 0.10 as can be seen in the figure below:

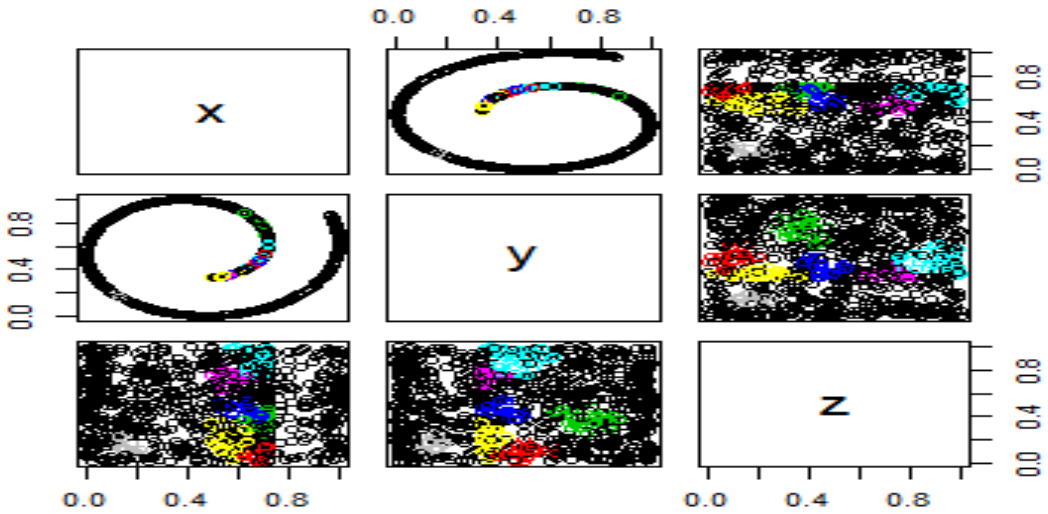


Calculation of minPts:

The default value of minPts is 5. When I ran the algorithm with value 5, I got around 15 clusters. So, with trial and error method, I got minPts value 17. I got 8 clusters for the dataset1 with that value.

Results:

I generated the results using two packages fpc and dbscan. I compared the results of both for equality for internal testing. I got the following plot on applying dbscan on dataset1.



Graph-Based clustering:

Graph based clustering starts with the proximity matrix. For Graph Based clustering, I used inbuilt package jpclust.

Parameter Selection:

Jpclust takes as input data matrix. The important part in the parameter selection is the value of k and kt.

K is Neighborhood size for nearest neighbor sparsification. If x is a kNN object, then k may be missing.

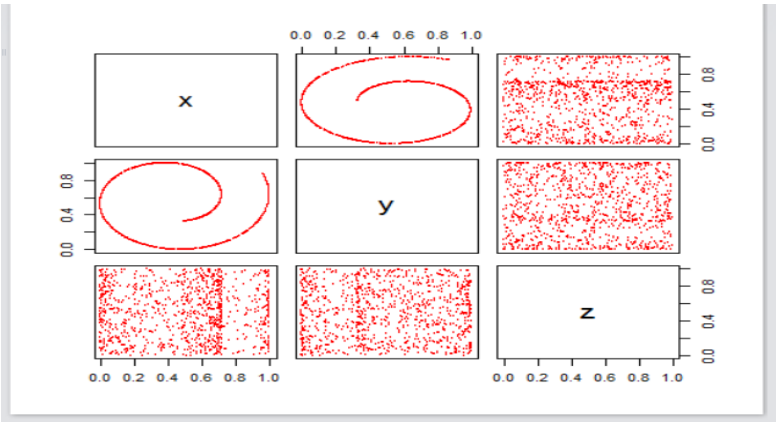
Kt is threshold on the number of shared nearest neighbors (including the points themselves) to form clusters.

Based on trial and error, I got value of k as 20 and kt = 8 based on the number of clusters.

Remove noise:

To remove noise from the plot created by the jpclust, I first applied Knn method on the data set and then used the object obtained from Knn to feed to the method jpclust to get efficient results.

Results:



Performance Analysis of different clustering techniques:

Out of the four clustering techniques, hierarchical and k-means provide the best results. They achieve the best clustering with sizes of each cluster: 115 97 99 114 125 122 169 159. The ground truth values given in the dataset1.csv is 150 for each cluster.

On the other hand, we can get 8 clusters in the density based clustering by finding the optimal value of eps and minPts. The result is affected by the noise data. Out of 1000 data sets, around 700 is considered as the noise data and clustered under the value 0.

Graph based clustering does not provide efficient results also. Even after applying the techniques to resolve noise data, there is not much change in the performance of the algorithm.

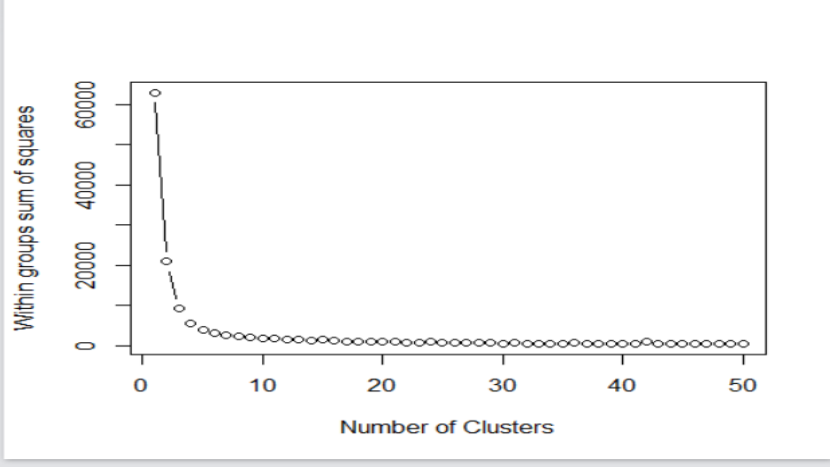
Clustering techniques on Dataset2:

I have chosen K means clustering method for dataset 2. The main challenge is to find the optimal value for K using wss.

Finding optimal value of clusters:

A K-Means Clustering algorithm allows us to group observations near the mean. This allows us to create greater efficiency in categorizing the data into specific segments. Here is how the wss command is used to firstly determine the number of clusters, and how kmeans is then used to conduct the k-means analysis.

For the analysis, I used maximum 20 clusters. Once this is done, we then identify the ideal number of clusters at the point at which there is a cut off in the plot:



From the above, we see that our scree plot cuts off at 3 clusters. This is the point at which the "within groups sum of squares" is minimized at the third cluster.

K Means Clustering:

One way is to run kmeans on the data with 3 clusters.

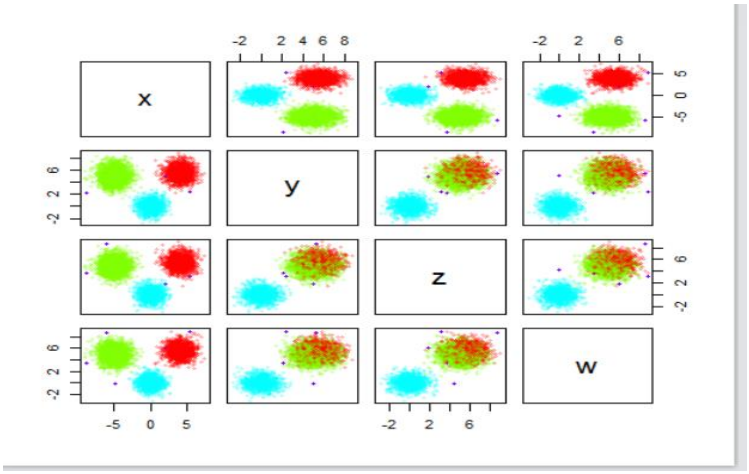
```
fit <- kmeans(data, 3)
```

Another possible solution which I have implemented in the script file is explained as below:

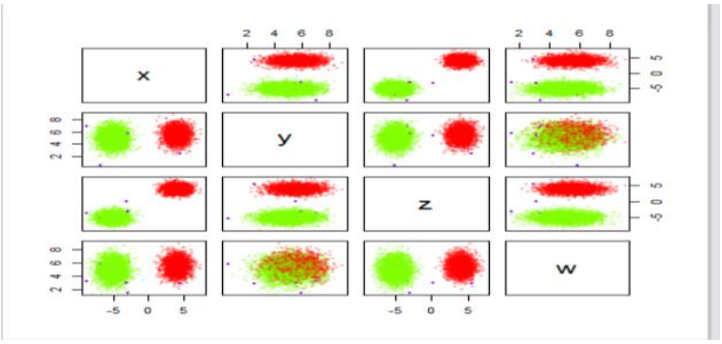
The solution is to sample the data, cluster the smaller sample, then treat the clustered sample as training data for k Nearest Neighbors and "classify" the rest of the data. I use a sample of 5000 points. The original data is not well-separated, but with only 1/220 of the data, the sample is separated.

Results obtained by this method is described as below:

As we can see, the plot shows three clusters as was calculated by optimal number of clusters method.



If we vary the value of K, we can obtain different clustering like two clusters for the same data set.



Performance analysis:

We need to find K (number of clusters) for which within sum of squares should be minimum. Hence, as we can see from the figure, that value of K for which wss is minimum, is 3. Hence, the best clustering will be performed when data set will be divided into 3 clusters.

Density and graph based time outs when run on dataset2, hence did not get any results for those two.

Conclusion:

Out of the four clustering techniques, K Means works best for both small and large datasets. Depending on the ideal value of K, we can significantly improve the results of the clustering process.

Density and graph based clustering techniques are based on finding the optimal value of eps.

References:

1. http://www.sthda.com/english/wiki/print.php?id=246
2. https://cran.r-project.org/web/packages/dbscan/dbscan.pdf
3. https://cran.r-project.org/web/packages/dendextend/vignettes/Cluster_Analysis.html
4. http://uc-r.github.io/kmeans_clustering
5. https://www.rdocumentation.org/packages/dbscan/versions/1.1-1/topics/jpclust