IDM Project 2
Clustering Report

Aniketh Sukhtankar (UF ID 7819 9584)

Introduction

The goal of the project is to increase familiarity with the clustering packages, available in R to do data mining analysis on real-world problems. Several different clustering methods were used on the given datasets. The dataset was as provided by the professor. The original cluster column was used as initial label for comparison. kMeans, Hierarchical, DBScan and SNNClust were the clustering methods used on the data set 1 and kMeans was chosen for data set 2. The following steps were run to get the desired results.

Clustering Methods Used

k-means clustering (k-means)

k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.
The problem is computationally difficult (NP-hard); however, there are efficient heuristic algorithms that are commonly employed and converge quickly to a local optimum. These are usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Additionally, they both use cluster centers to model the data; however, k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes.
The algorithm has a loose relationship to the k-nearest neighbor classifier, a popular machine learning technique for classification that is often confused with k-means because of the k in the name. One can apply the 1-nearest neighbor classifier on the cluster centers obtained by k-means to classify new data into the existing clusters. This is known as nearest centroid classifier or Rocchio algorithm.
In my script I use kmeans function to get an effective clustering of the data. The centers parameter is set to 8 to get the 8 clusters as desired. The nstart value is set to 25 to choose 25

random sets for centers. External validation was used with adjusted rand index method to get the summary statistics for the given clustering method. Finally, the 3D plots for original and clustered data were plotted using the scatter3d function, original labels and cluster result label values.

Hierarchical clustering

Hierarchical clustering groups data over a variety of scales by creating a cluster tree or dendrogram. The tree is not a single set of clusters, but rather a multilevel hierarchy, where clusters at one level are joined as clusters at the next level. This allows you to decide the level or scale of clustering that is most appropriate for your application. The function hclust supports agglomerative clustering and performs all of the necessary steps for you. The fviz_dend function in factoextra plots the cluster tree.
The following procedure is followed:
1. The similarity or dissimilarity between every pair of objects in the data set is calculated. In this step, you calculate the distance between objects using the dist function. The dist function supports many different ways to compute this measurement. We use Euclidean as our method of choice.
2. Group the objects into a binary, hierarchical cluster tree. In this step, you link pairs of objects that are in close proximity using the hclust function. Ward's minimum variance method which was used aims at finding compact, spherical clusters. The hclust function uses the distance information generated in step 1 to determine the proximity of objects to each other. As objects are paired into clusters, the newly formed clusters are grouped into larger clusters until a hierarchical tree is formed. See hclust for more information.
3. Determine where to cut the hierarchical tree into clusters. In this step, you use the cutree function to prune branches off the bottom of the hierarchical tree, and assign all the objects below each cut to a single cluster. This creates a partition of the data. The cutree function can create these clusters by detecting natural groupings in the hierarchical tree or by cutting off the hierarchical tree at an arbitrary point.

External validation was used with adjusted rand index method to get the summary statistics for the given clustering method. Finally, the 3D plots for original and clustered data were plotted using the scatter3d function, original labels and cluster result label values.

Density-based clustering

Density-based spatial clustering of applications with noise (DBSCAN) is the density-based clustering algorithm used: given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie

alone in low-density regions (whose nearest neighbors are too far away). DBSCAN is one of the most common clustering algorithm. DBSCAN requires two parameters: ε (eps) and the minimum number of points required to form a dense region (minPts). It starts with an arbitrary starting point that has not been visited. This point's ε-neighborhood is retrieved, and if it contains sufficiently many points, a cluster is started. Otherwise, the point is labeled as noise. Note that this point might later be found in a sufficiently sized ε-environment of a different point and hence be made part of a cluster. If a point is found to be a dense part of a cluster, its ε-neighborhood is also part of that cluster. Hence, all points that are found within the ε-neighborhood are added, as is their own ε-neighborhood when they are also dense. This process continues until the density-connected cluster is completely found. Then, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise. DBSCAN can be used with any distance function (as well as similarity functions or other predicates). The distance function (dist) can therefore be an additional parameter.

In my script I use kNNdistplot that does fast calculation of the k-nearest neighbor distances in a matrix of points. The plot was used by me to help find a suitable value for the eps neighborhood for DBSCAN. This was done by looking for the knee in the plot. The eps value of 1.463 was found to be optimal. The fpc::dbscan function was used to get an effective clustering of the data. The MinPts parameter is set to 8 to get the 8 clusters as desired. The eps value is set to 1.463 as suggested by the kNNdistplot. External validation was used with adjusted rand index method to get the summary statistics for the given clustering method. Finally, the 3D plots for original and clustered data were plotted using the scatter3d function, original labels and cluster result label values.

Graph-based clustering

Implemented the shared nearest neighbor clustering algorithm by Ertoz, Steinbach and Kumar. Algorithm: 1) Constructs a shared nearest neighbor graph for a given k. The edge weights are the number of shared k nearest neighbors (in the range of [0, k]). 2) Find each points SNN density, i.e., the number of points which have a similarity of eps or greater. 3) Find the core points, i.e., all points that have an SNN density greater than MinPts. 4) Form clusters from the core points and assign border points (i.e., non-core points which share at least eps neighbors with a core point). Note that steps 2-4 are DBSCAN and that eps is used on a similarity (the number of shared neighbors) and not on a distance like in DBSCAN.

In my script I use kNNdistplot that does fast calculation of the k-nearest neighbor distances in a matrix of points. The plot was used by me to help find a suitable value for the eps neighborhood for SNN. This was done by looking for the knee in the plot. The eps value of 1.8 was found to be optimal. The dbscan::sNNclust function was used to get an effective clustering of the data. The

MinPts parameter is set to 10 to get the 8 clusters as desired. The eps value is set to 1.8 as suggested by the kNNdistplot. Further the k value is set to 10. External validation was used with adjusted rand index method to get the summary statistics for the given clustering method. Finally, the 3D plots for original and clustered data were plotted using the scatter3d function, original labels and cluster result label values.

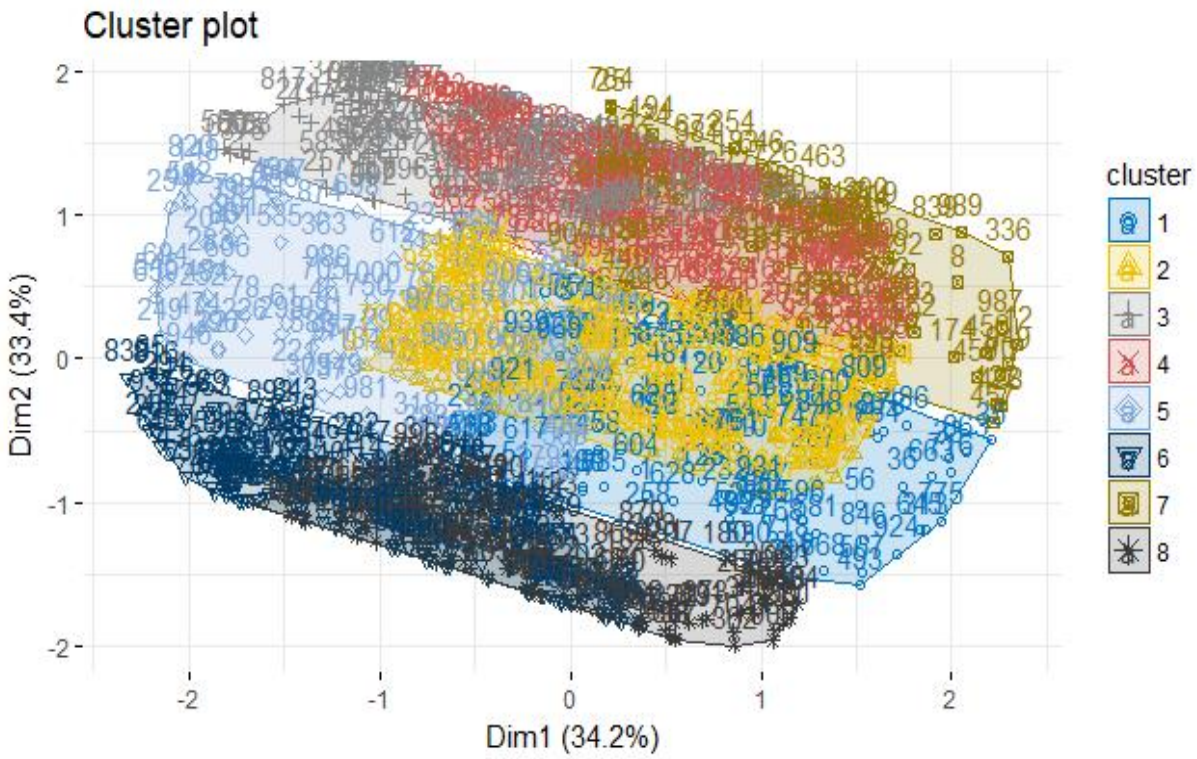
Clustering Results and Analysis (Dataset 1)

k-means clustering (k-means)

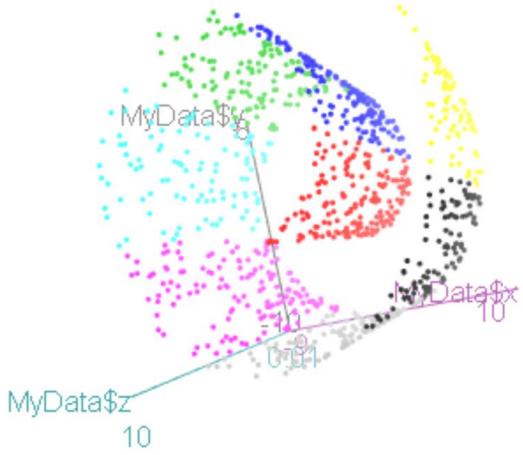
1. SAMPLE 1 (Seed: 123)

The final values used for the model was centers = 8, nstart = 25.

The following plot was obtained using factoextra.



```
----------------------------------------
purity                          : 0.166
entropy                         : 0.9733
normalized mutual information   : 0.0131
variation of information        : 5.8808
normalized var. of information  : 0.9934
----------------------------------------
specificity                     : 0.8688
sensitivity                     : 0.1316
precision                       : 0.1245
recall                          : 0.1316
F-measure                       : 0.128
----------------------------------------
accuracy OR rand-index          : 0.7773
adjusted-rand-index             : 5e-04
jaccard-index                   : 0.0684
fowlkes-mallows-index           : 0.128
mirkin-metric                   : 222450
----------------------------------------
```



Original Cluster Scatterplot

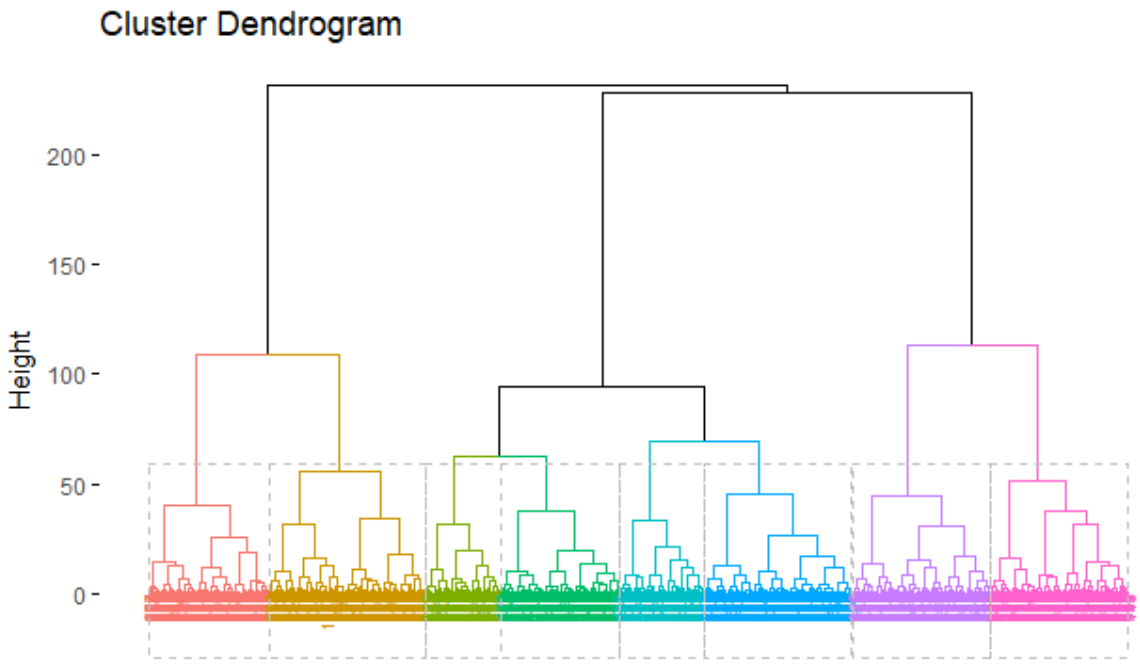
Scatterplot after k means Clustering

Run for seeds 1234,1111,2222 and 3333 with accuracies 0.7778,0.7778,0.7773 and 0.7773.
Average Accuracy: 0.7775

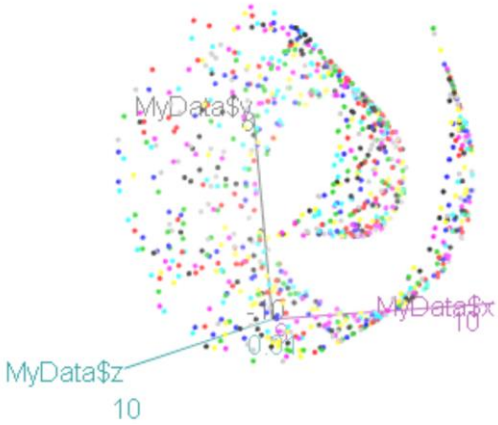
Hierarchical Clustering

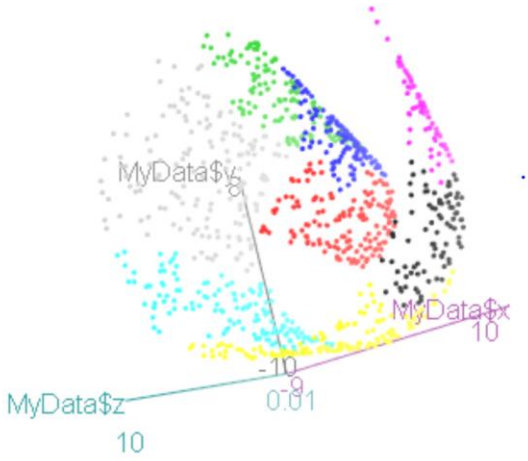
1. SAMPLE 1 (Seed: 123)

The following dendogram was obtained using factoextra.




```
----------------------------------------
purity                          : 0.167
entropy                         : 0.9755
normalized mutual information   : 0.0122
variation of information        : 5.8901
normalized var. of information : 0.9939
----------------------------------------
specificity                     : 0.8699
sensitivity                     : 0.1305
precision                       : 0.1244
recall                          : 0.1305
F-measure                       : 0.1274
----------------------------------------
accuracy OR rand-index          : 0.7781
adjusted-rand-index             : 3e-04
jaccard-index                   : 0.068
fowlkes-mallows-index           : 0.1274
mirkin-metric                   : 221698
----------------------------------------
```





Original Cluster Scatterplot Scatterplot after Hierarchical Clustering

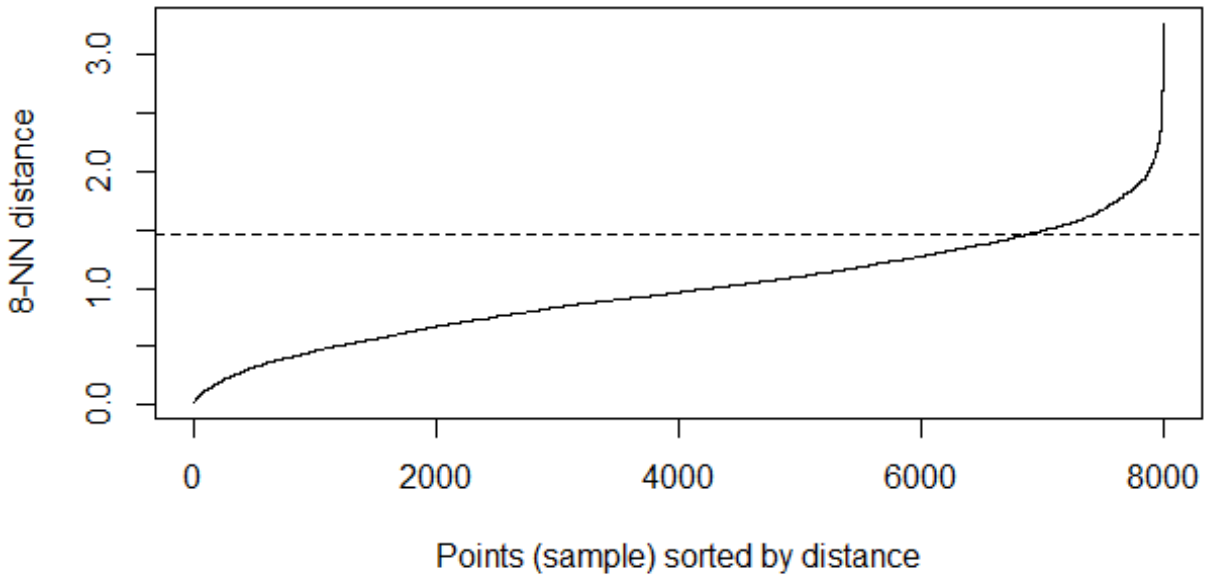
Run for seeds 1234, 1111, 2222 and 3333 with accuracies 0.7781, 0.7781, 0.7781 and 0.7781.

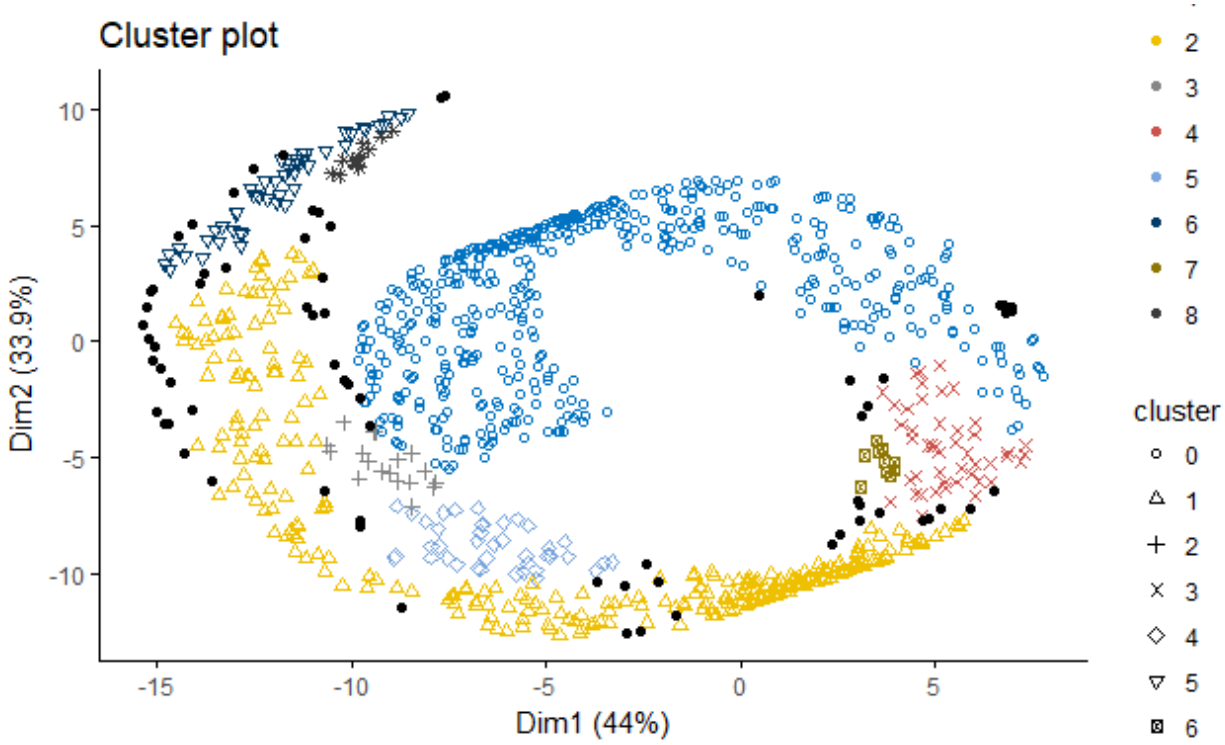
Average Accuracy: 0.7781

Density-based clustering

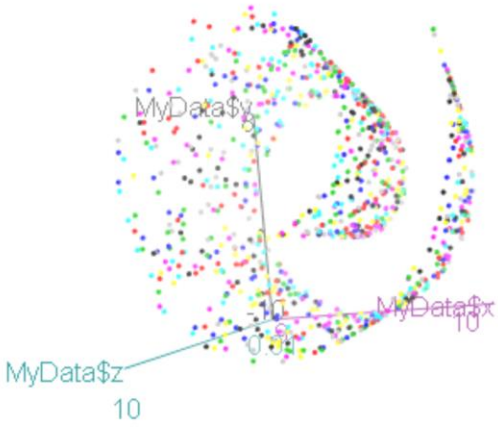
1. SAMPLE 1 (Seed: 123)

The final values used for the model were eps = 1.463, MinPts = 8.

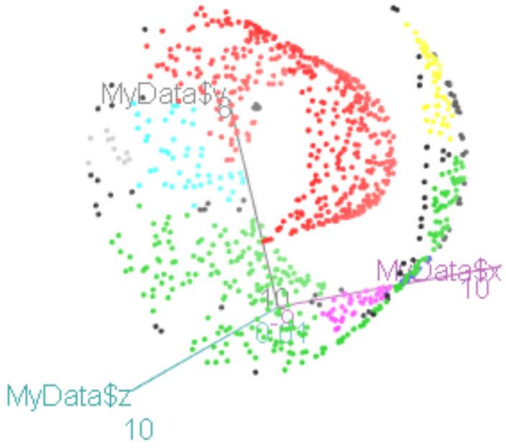





```
----------------------------------------
purity                          : 0.164
entropy                         : 0.6978
normalized mutual information   : 0.0208
variation of information        : 5.0399
normalized var. of information  : 0.9895
----------------------------------------
specificity                     : 0.6861
sensitivity                     : 0.3152
precision                       : 0.1246
recall                          : 0.3152
F-measure                       : 0.1785
----------------------------------------
accuracy OR rand-index          : 0.64
adjusted-rand-index             : 7e-04
jaccard-index                   : 0.098
fowlkes-mallows-index           : 0.1981
mirkin-metric                   : 359596
----------------------------------------
```



Original Cluster Scatterplot



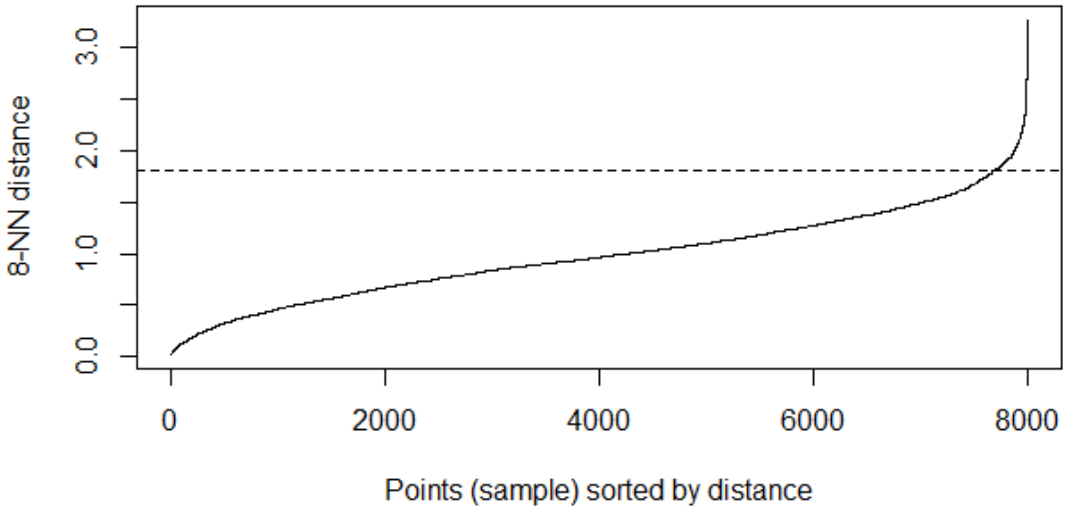
Scatterplot after DBScan Clustering

Similarly Run for seeds 1234, 1111, 2222 and 3333 with accuracies 0.64, 0.64, 0.64 and 0.64.
Average Accuracy: 0.64

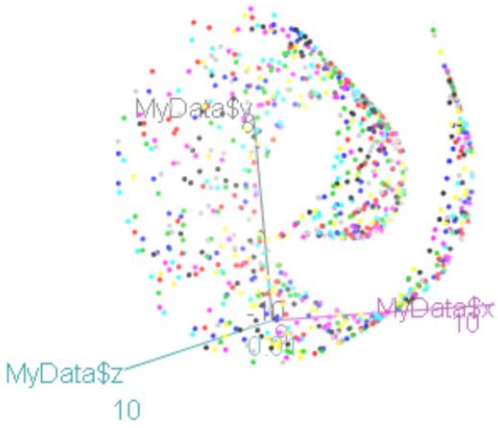
Graph-based clustering

1. SAMPLE 1 (Seed: 123)

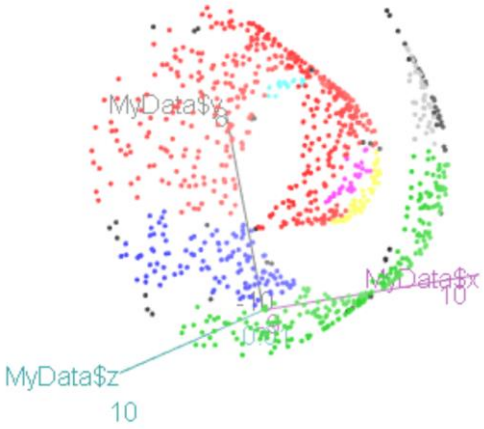
The final values used for the model were k = 10, eps = 1.8, minPts = 10, borderPoints = TRUE.



```
----------------------------------------
purity                          : 0.165
entropy                         : 0.7573
normalized mutual information   : 0.0183
variation of information        : 5.2232
normalized var. of information : 0.9907
----------------------------------------
specificity                     : 0.7208
sensitivity                     : 0.2826
precision                       : 0.1254
recall                          : 0.2826
F-measure                       : 0.1738
----------------------------------------
accuracy OR rand-index          : 0.6664
adjusted-rand-index             : 0.0022
jaccard-index                   : 0.0951
fowlkes-mallows-index           : 0.1883
mirkin-metric                   : 333262
----------------------------------------
```



Original Cluster Scatterplot



Scatterplot after Graph-based Clustering

Similarly Run for seeds 1234, 1111, 2222 and 3333 with accuracies 0.6664, 0.6664, 0.6664 and 0.6664

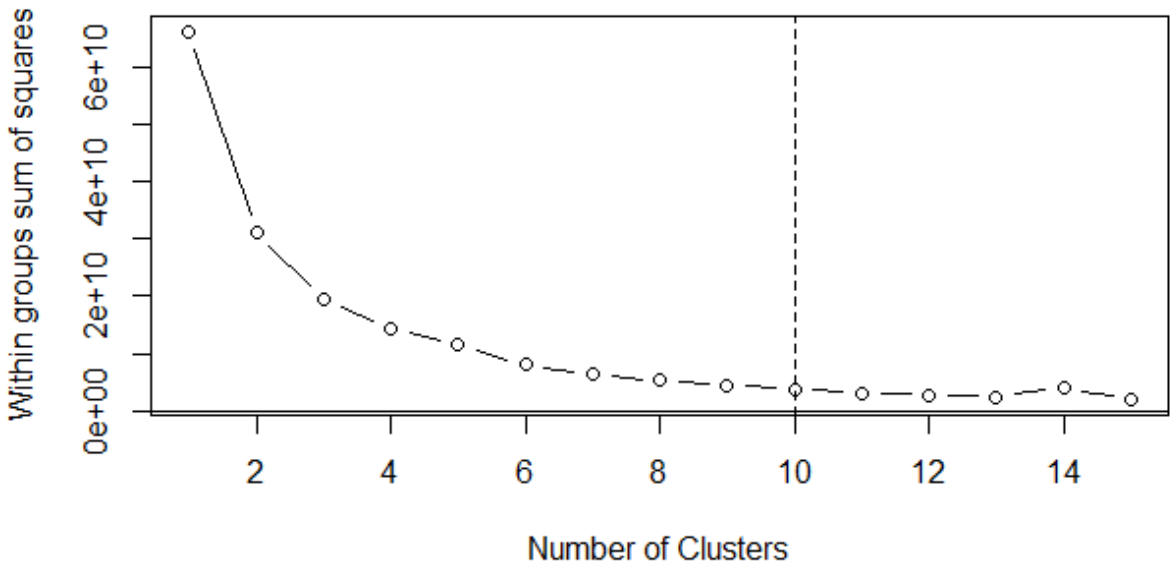
Average Accuracy: 0.6664

Clustering Results and Analysis (Dataset 2)

The second dataset, named as dataset2.csv contained more than 1 million data points with the coordinates in 4 dimensions. K-means Clustering was chosen from the above 4 clustering techniques and applied on this dataset. The following factors were considered:

• Optimal No. of clusters to seek: Recall that, the basic idea behind partitioning methods, such as k-means clustering is to define clusters such that the total intra-cluster variation [or total within-cluster sum of square (WSS)] is minimized. The total WSS measures the compactness of the clustering and we want it to be as small as possible.

The Elbow method looks at the total WSS as a function of the number of clusters: One should choose a number of clusters so that adding another cluster doesn't improve much better the total WSS.
The optimal number of clusters were calculated as follows:
1. Computation was done for k-means clustering algorithm for different values of k. For instance, by varying k from 2 to 15 clusters in my case.
2. For each k, the total within-cluster sum of square (wss) was calculated.
3. The curve of wss according to the number of clusters k was plotted as given below.
4. The location of a bend (knee) in the plot which is 10 in my case is generally considered as an indicator of the appropriate number of clusters.



• The best method that can be applied for this case:

One of the main advantages of K-Means is that it is the fastest partitional method for clustering large data that would take an impractically long time with similar methods.

If you compare the time complexities of K-Means with other methods: K-Means is O(tkn), where n is the number of objects, k is the number of clusters, and t is how many iterations it takes to converge. K-Medoids is O(k(n−k)2) for each iteration. Agglomerative hierarchical clustering is O(n3) (more efficient agglomerative clustering techniques are O(n2)), while exhaustive divisive hierarchical clustering is O(2n), so these algorithms don't scale well for large datasets.

However, K-Means has the following drawbacks:

You need to specify the number of clusters in advanced (you could try a range of numbers, or use mathematical methods to calculate the optimal number like the elbow method used above).
It struggles to find clusters which don't have a roughly spherical shape
It struggles to find clusters of vastly different sizes
It is less able to handle noise/outliers compared to other methods.

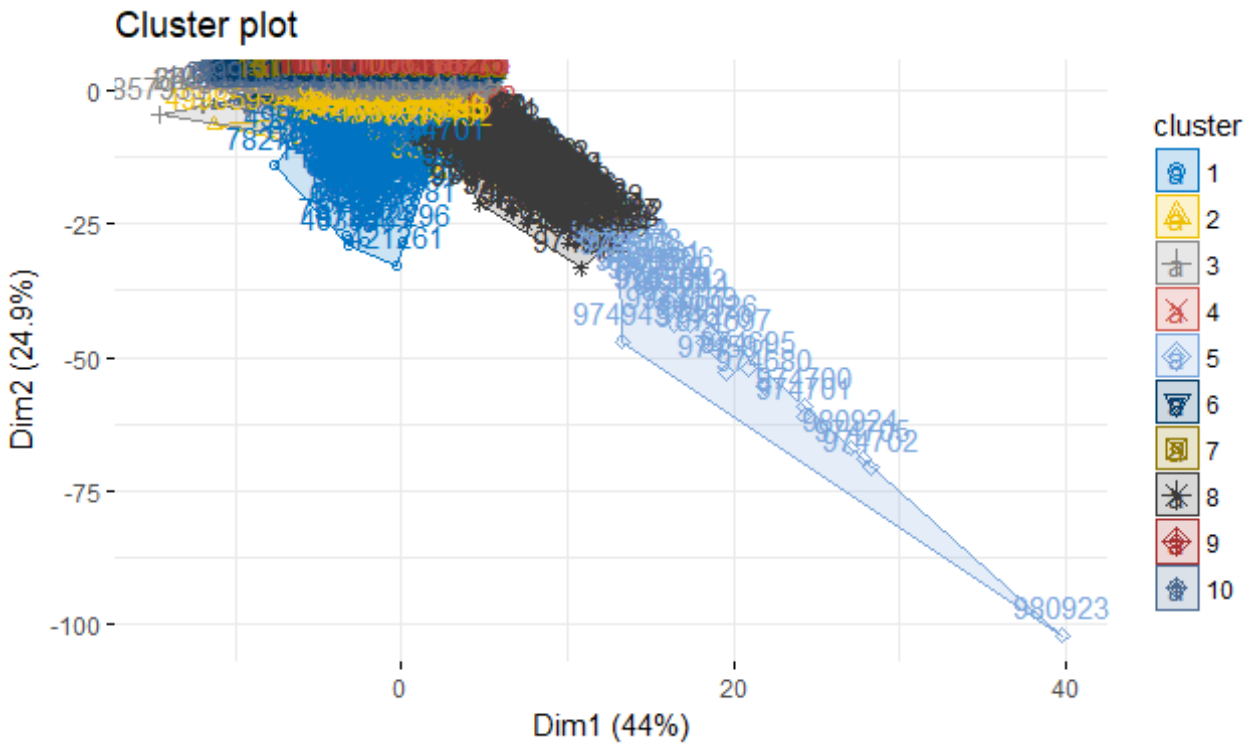
Hierarchical clustering algorithms like CURE and BIRCH as well as K-means algorithm work well with large datasets. Among the clustering techniques that I have implemented, K-means worked best for the given dataset. Further, the Elbow method for determining the optimal number of clusters incorporates K-means in its computation, thereby ensuring cluster compactness if K-means algorithm is chosen.

You can also use other methods if you take a sample of the data

The methods which do not work well with large datasets are often more accurate than K-Means (which is why they take longer). You can use these methods if you take a sample of the large dataset. You could take a random sample, or a biased sample which exaggerates denser areas. CLARA is an option for sampling a large dataset to use K-Medoids on.

Evaluation

The following cluster plot was obtained using factoextra.



In order to do a measure of the "goodness" of the k-means clustering I calculated the (Between Sum of squares) BSS/TSS (Total Sum of squares) ratio. SS stands for Sum of Squares, so it's the usual decomposition of deviance in deviance "Between" and deviance "Within". Ideally you want a clustering that has the properties of internal cohesion and external separation, i.e. the BSS/TSS ratio should approach 1. For eg. The iris Dataset gives a BSS/TSS ratio of 88.4% (0.884) indicating a good fit. The ratio for our dataset was 94.5% (0.9449151) which indicates a good fit.

Between Sum of Squares: 62675316058

Total Sum of Squares: 66329045576

BSS/TSS ratio: 0. 9449151

Conclusion

This project was mainly concerned with performing basic clustering algorithms such as k-means, Hierarchical, DBScan and SNNClust with the help of R. The data set 1 that was used was small and over viewable. The method for tuning the parameters of various algorithms determines the clustering accuracies and number of clusters formed.

The following are the cumulated results obtained for each of the algorithms for all the five seed values in the form of average and standard deviation. As we can see below, Hierarchical and kmeans were the most accurate among all the clustering methods used with an accuracy of approximately 78%. K-means Clustering was chosen for dataset 2 and the Elbow method was used to calculate optimal no. of clusters to be used. The BSS/TSS ratio for dataset 2 was 94.5% (0.9449151) which indicates a good fit.

	Mean	Standard Deviation
K Means	0.7775	0.00027
Hierarchical	0.7781	0
DBScan	0.64	0
SNNClust	0.6664	0

References

- https://www.r-bloggers.com/k-means-clustering-on-big-data/
- https://www.statmethods.net/advstats/cluster.html
- http://www.sthda.com/english/articles/25-cluster-analysis-in-r-practical-guide/111-types-of-clustering-methods-overview-and-quick-start-r-code/
- http://www.sthda.com/english/wiki/amazing-interactive-3d-scatter-plots-r-software-and-data-visualization
- https://www.r-bloggers.com/finding-optimal-number-of-clusters/
- https://stats.stackexchange.com/questions/183197/k-means-algorithm-for-big-data-analytics/
- http://dkmathstats.com/the-k-means-algorithm-in-r-for-clustering-data/
- https://stats.stackexchange.com/questions/82776/what-does-total-ss-and-between-ss-mean-in-k-means-clustering/82779/