Introduction to Data Mining (CIS 6930)

Fall 2017

Project 2: Clustering Report

Shikha Dharmendra Mehta

UFID 48519256

shikha.mehta@ufl.edu

DATASET PREPARATION & OVERVIEW

Dataset 1 (given) is used in the first part of the project. After importing into R, the variable for ground-truth labels (cluster) is separated from the dataset, and the main data is used further for clustering.

Agglomerative Hierarchical, K-means, DBSCAN (i.e. Density-based) and Shared Nearest Neighbor (i.e. Graph-based) were the clustering methods run on dataset 1. Adjusted Random Index mapping strategy was used to compare the predicted cluster values with the ground-truth cluster labels.

Dataset 2 (given) is used in the second part of the project. After importing into R, we view the dataset as a wss-plot (Within groups sum of squares vs Number of clusters) to determine the optimal number of clusters (k) for the data. The value of k is chosen by interpreting the plot.

The K-means clustering algorithm was then run on the dataset with the appropriate value of k. Results were evaluated by measuring Total Variance, given by the ratio of Between Sum of squares (BSS) to Total Sum of squares (TSS).

CLUSTERING TECHNIQUES

1. Hierarchical Clustering (Agglomerative Hierarchical)

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:

i. 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.

ii. 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.

iii. 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 function 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.

2. 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 function 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.

3. Density-based Clustering (DBSCAN)

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 10 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 10 as suggested by the kNNdistplot. External_validation function 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.

4. Graph-based Clustering (Shared Nearest Neighbor)

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 10 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 function 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.

DATASET 1 - DETAILED ANALYSIS

Summary of Entire Dataset

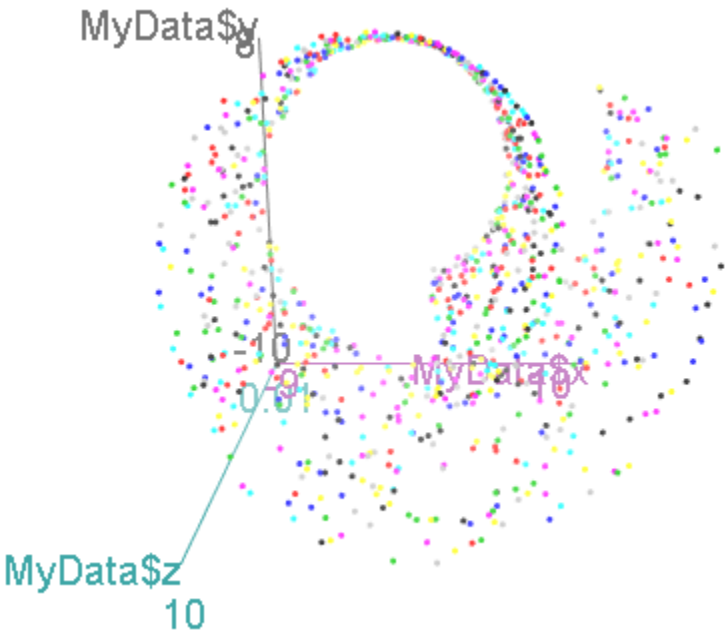
Total Number of Observations	1000
Total Number of Feature Variables	3
Total Number of Ground Truth Labels	8

Accuracies (Average) for all Clustering Techniques

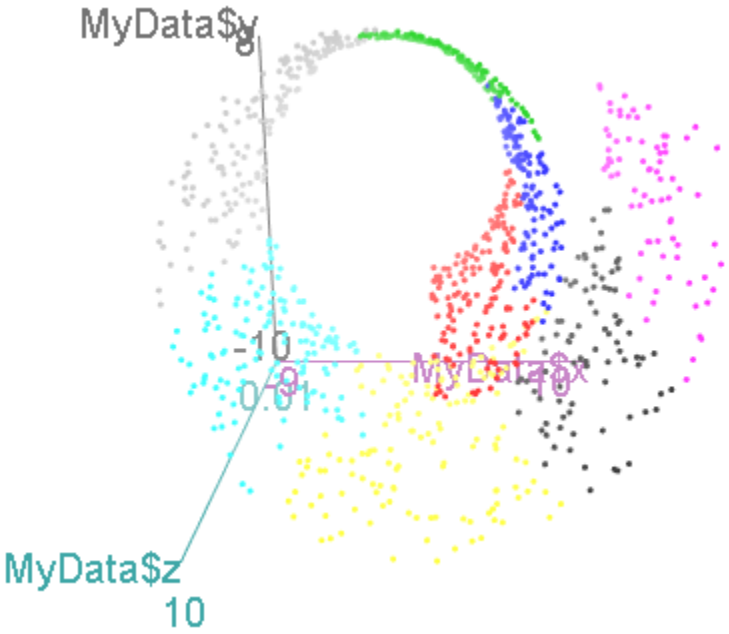
	Hierarchical	K-means	DBSCAN	Shared Nearest Neighbor
Accuracy	0.7781	0.7776	0.64	0.7308

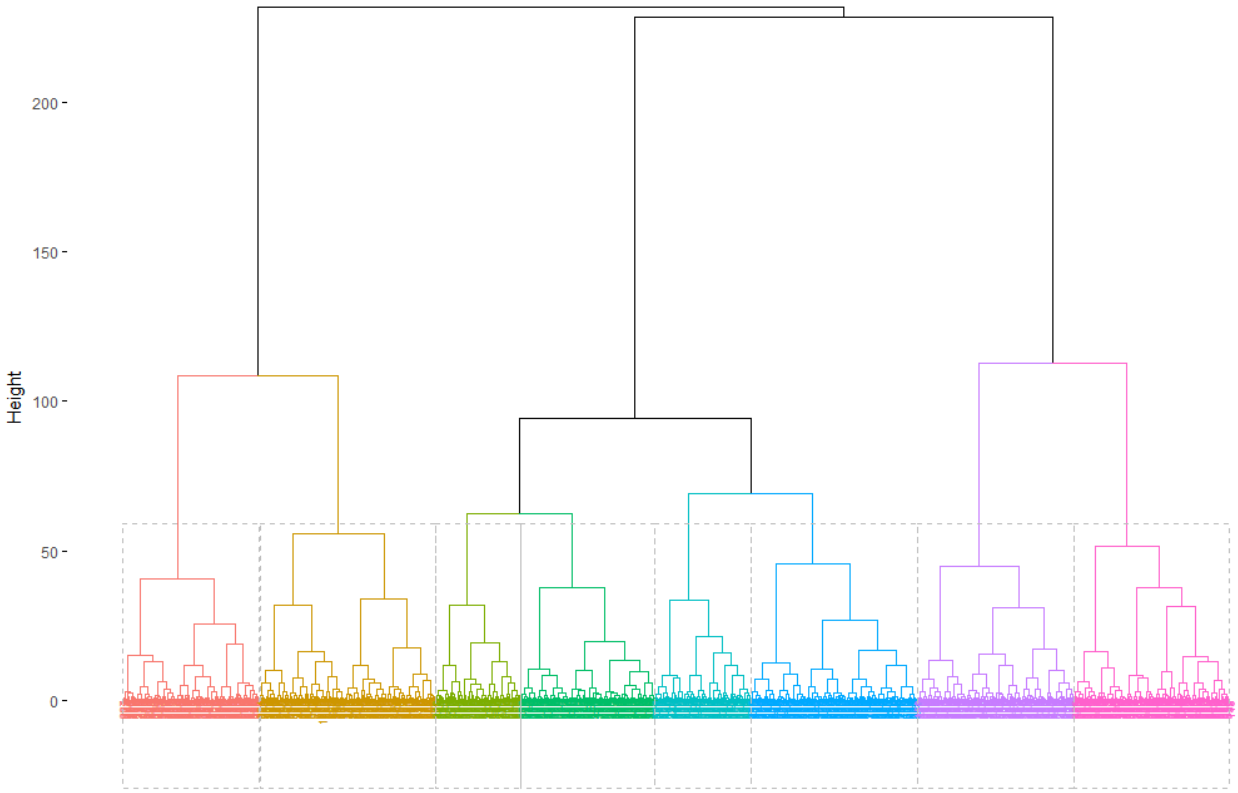
Note that multiple iterations only affected K-means clustering, since it involves randomly selecting the initial centroids. For the other three clustering algorithms, the results generated were the same across iterations.

3D Graphic Plot of the Actual Data



1. Agglomerative Hierarchical Clustering – 3D Graphic Plot and other metrics

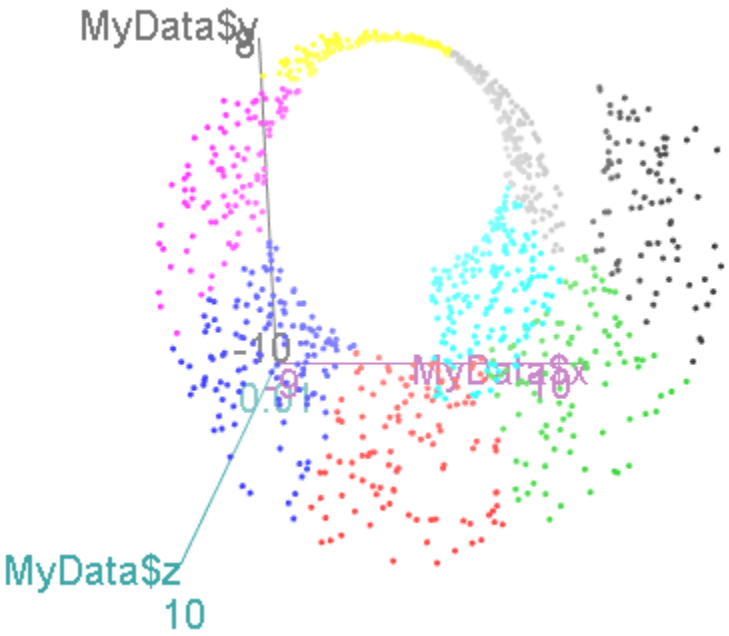




```
----------------------------------------
purity                          : 0.167
entropy                         : 0.9755
normalized mutual information   : 0.0122
variation of information        : 5.8901
normalized var. of information  : 0.9939
----------------------------------------
specificity                     : 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
----------------------------------------
```

2. K-means Clustering – 3D Graphic Plot and other metrics

Iteration 1 (Seed Value - 231):




```
----------------------------------------
purity                          : 0.163
entropy                         : 0.9754
normalized mutual information   : 0.0127
variation of information        : 5.8882
normalized var. of information  : 0.9936
----------------------------------------
specificity                     : 0.8696
sensitivity                     : 0.1305
precision                       : 0.1242
recall                          : 0.1305
F-measure                       : 0.1273
----------------------------------------
accuracy OR rand-index          : 0.7778
adjusted-rand-index             : 1e-04
jaccard-index                   : 0.068
fowlkes-mallows-index           : 0.1273
mirkin-metric                   : 221932
----------------------------------------
```

Iteration 2 (Seed Value - 232):

```
----------------------------------------
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
----------------------------------------
```

Iteration 3 (Seed Value - 233):

```
----------------------------------------
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
----------------------------------------
```

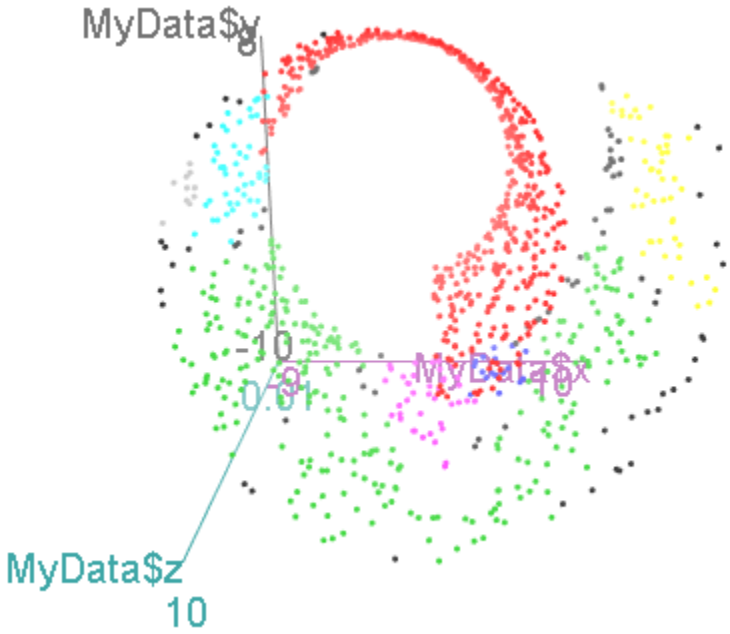
Iteration 4 (Seed Value - 234):

```
----------------------------------------
purity                          : 0.163
entropy                         : 0.9754
normalized mutual information   : 0.0127
variation of information        : 5.8882
normalized var. of information  : 0.9936
----------------------------------------
specificity                     : 0.8696
sensitivity                     : 0.1305
precision                       : 0.1242
recall                          : 0.1305
F-measure                       : 0.1273
----------------------------------------
accuracy OR rand-index          : 0.7778
adjusted-rand-index             : 1e-04
jaccard-index                   : 0.068
fowlkes-mallows-index           : 0.1273
mirkin-metric                   : 221932
----------------------------------------
```

Iteration 5 (Seed Value - 235):

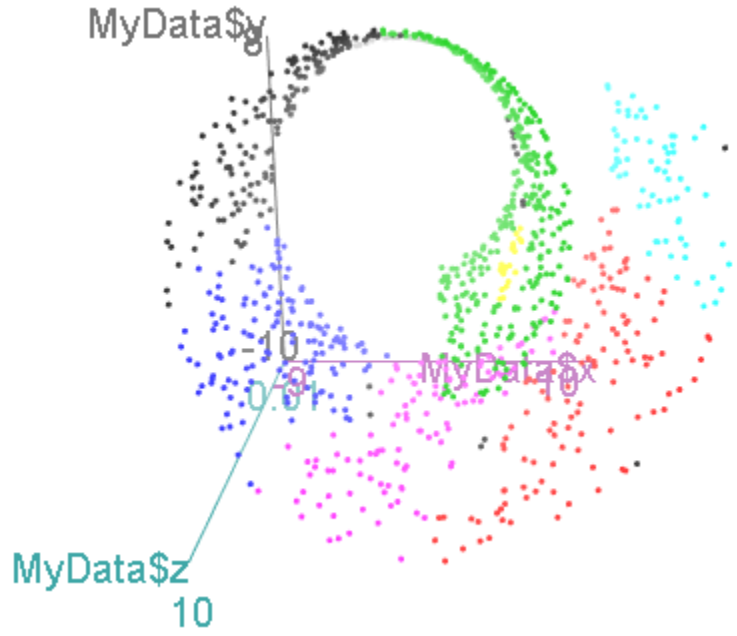
```
----------------------------------------
purity                          : 0.163
entropy                         : 0.9754
normalized mutual information   : 0.0127
variation of information        : 5.8882
normalized var. of information  : 0.9936
----------------------------------------
specificity                     : 0.8696
sensitivity                     : 0.1305
precision                       : 0.1242
recall                          : 0.1305
F-measure                       : 0.1273
----------------------------------------
accuracy OR rand-index          : 0.7778
adjusted-rand-index             : 1e-04
jaccard-index                   : 0.068
fowlkes-mallows-index           : 0.1273
mirkin-metric                   : 221932
----------------------------------------
```

3. DBSCAN (Density-based) Clustering – 3D Graphic Plot and other metrics




```
----------------------------------------
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
----------------------------------------
```

4. **Shared Nearest Neighbor (Graph-based) Clustering – 3D Graphic Plot and other metrics**




```
----------------------------------------
purity                          : 0.157
entropy                         : 0.8796
normalized mutual information   : 0.0137
variation of information        : 5.5998
normalized var. of information  : 0.9931
----------------------------------------
specificity                     : 0.8071
sensitivity                     : 0.1921
precision                       : 0.1237
recall                          : 0.1921
F-measure                       : 0.1505
----------------------------------------
accuracy OR rand-index          : 0.7308
adjusted-rand-index             : -6e-04
jaccard-index                   : 0.0814
fowlkes-mallows-index           : 0.1542
mirkin-metric                   : 268970
----------------------------------------
```

DATASET 2 - DETAILED ANALYSIS

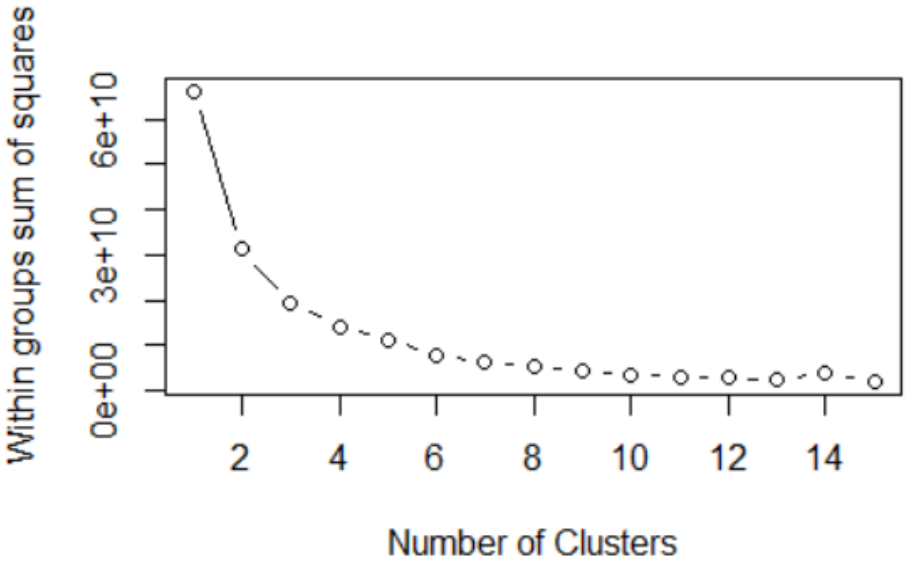
Summary of Entire Dataset

Total Number of Observations	1006183
Total Number of Feature Variables	4

From the clustering techniques described in previous sections, **K-means Clustering** was chosen to perform clustering on this dataset by considering the following factors:

- *The optimal number of clusters to seek*

  The basic idea behind data 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. Thus, the total WSS measures the compactness of the clustering and we want it to be as small as possible for a given dataset.



  The Elbow method looks at the total WSS as a function of the number of clusters, i.e., one should choose a number of clusters so that adding another cluster doesn't improve the total WSS much. The optimal number of clusters are then calculated as follows:

1. Computation was done for k-means clustering algorithm for different values of k. For my analysis, I varied k from 2 to 15 clusters.
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 is generally considered as an indicator of the appropriate number of clusters. This was found to be 10 in my case, as shown by the graph above.

- *The best method that can be applied for this case*

  Hierarchical clustering algorithms like CURE and BIRCH as well as K-means algorithm are commonly used with large datasets. Among the clustering techniques that I have implemented, K-means worked best for the given dataset.

  Another advantage 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. Agglomerative hierarchical clustering is $O(n^3)$ (more efficient agglomerative clustering techniques are $O(n^2)$), while exhaustive divisive hierarchical clustering is $O(2^n)$. Therefore, these algorithms don't scale well for large datasets.

  Also note that 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.

To evaluate the results of k-means clustering on the given data, I calculated the Total Variance, given by (Between Sum of squares) BSS/TSS (Total Sum of squares) ratio. Since we want a clustering that has the properties of internal cohesion and external separation, the value of Total Variance should approach 1. In the iris dataset for example, it is 0.884 (88.4%), indicating a good fit.

The Total Variance for our dataset was 0.9449151 (94.5%).
Between Sum of Squares: 62675316058
Total Sum of Squares: 66329045576
BSS/TSS ratio: 0. 9449151

CONCLUSION

The objectives of this project have been met. For Dataset 1, Agglomerative Hierarchical-based clustering gave the best average accuracy among the 4 chosen clustering algorithms, followed closely by K-means. In Dataset 2, we saw that K-means clustering algorithm with number of cluster centers = 10 was found to be a good fit for the data, with Total Variance close to 1. A couple of observations that were made are as follows:

- Tuning different parameters in the clustering algorithms affect the accuracies and the number of clusters formed.
- Running multiple iterations for K-means did not vary its accuracy by much.
- Many clustering algorithms failed when the size of the dataset was increased.

REFERENCES

[1] https://stats.stackexchange.com/questions/260229/comparing-a-clustering-algorithm-partition-to-a-ground-truth-one

[2] http://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation

[3] https://cran.r-project.org/web/packages/ClusterR/vignettes/the_clusterR_package.html

[4] https://www.r-bloggers.com/k-means-clustering-on-big-data/

[5] https://www.statmethods.net/advstats/cluster.html

[6] http://www.sthda.com/english/articles/25-cluster-analysis-in-r-practical-guide/111-types-of-clustering-methods-overview-and-quick-start-r-code/

[7] http://www.sthda.com/english/wiki/amazing-interactive-3d-scatter-plots-r-software-and-data-visualization

[8] https://www.r-bloggers.com/finding-optimal-number-of-clusters

[9] https://stats.stackexchange.com/questions/183197/k-means-algorithm-for-big-data-analytics/

[10] https://stats.stackexchange.com/questions/82776/what-does-total-ss-and-between-ss-mean-in-k-means-clustering/82779/