Report for Data Mining & Analytics Project

Group: 1A أحمد ياسر محمد محمد نصار

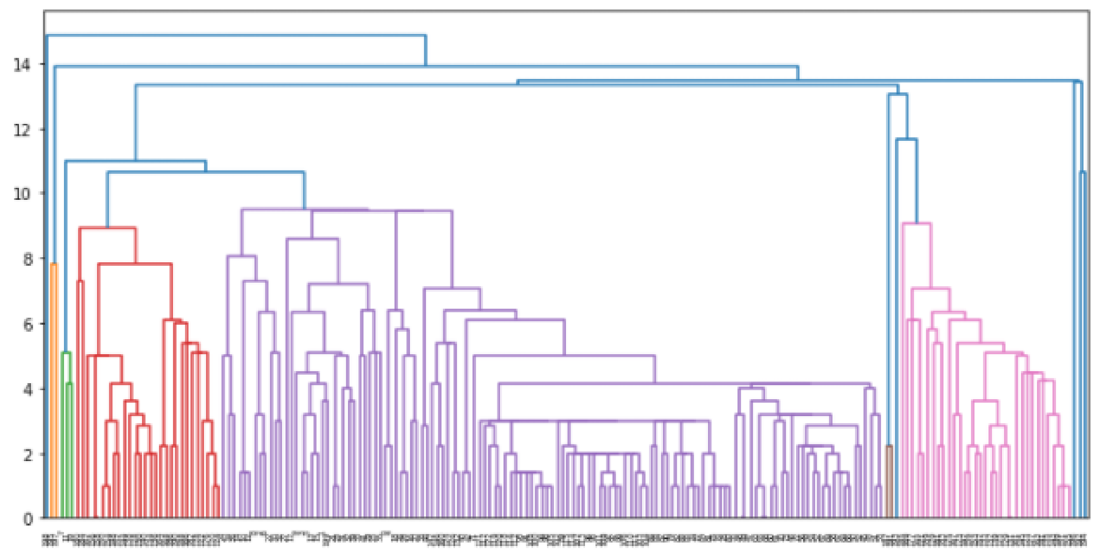
ID: 20191616316 Email: cds.AhmedYasser16316@alexu.edu.eg

This entire project was done solo by one member only including finding the dataset, coding & writing this report.

1) The Dataset link:
   Mall Customer Segmentation Data | Kaggle
2) This Dataset has 200 rows x 5 columns which are:
   (Customer ID, Gender, Age, Annual Income, Spending Score)
   My interest in this dataset will be for the two columns called (Annual Income & Spending Score) in addition to that I wanted to do experiment or test on another two columns which are:
   (Age, Annual Income) , In this data set the objective is to find the relationship or the pattern that customers have between their income by a year & their total money spent in malls by a scoring system , (The Target) is to Find how many groups of people have or belong to the same pattern and to explore the data as well by two clustering algorithms which are Agglomerative Hierarchal clustering & K Medoids clustering as well.

3) Visualization: (Agglomerative Hierarchal Clustering)

```
In [1043]: plt.figure(figsize=(12, 6))
           HierDend = sch.dendrogram(sch.linkage(Final,'single'))
```

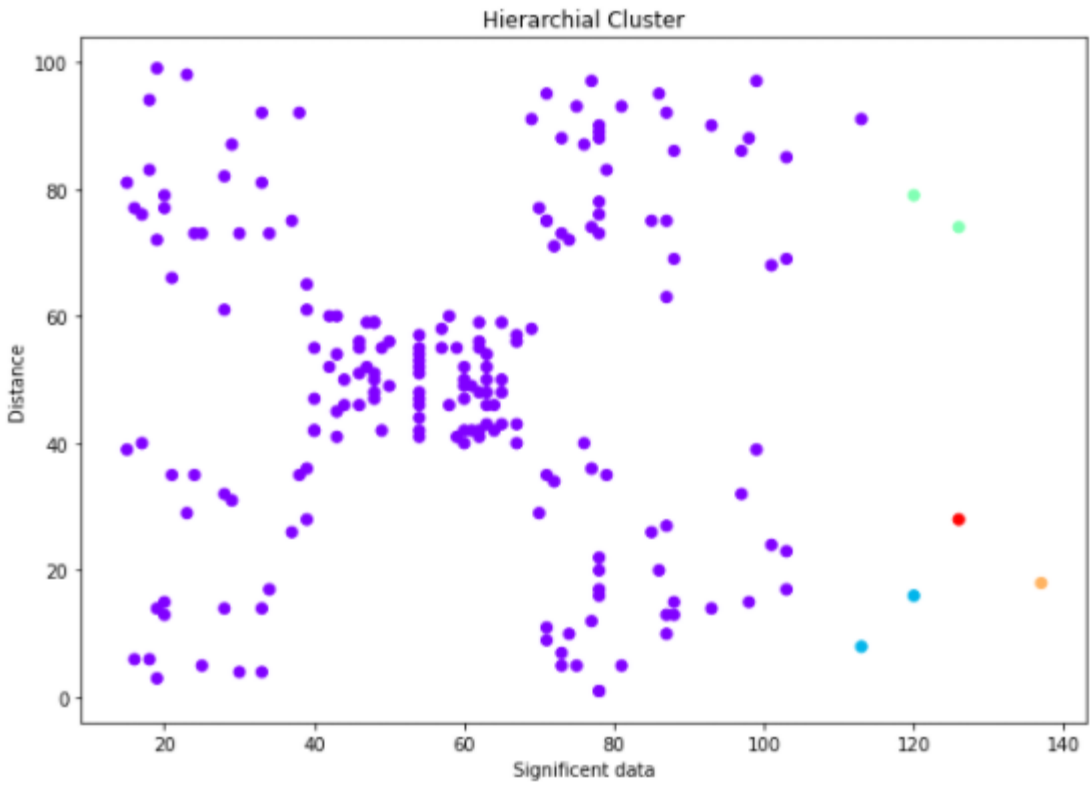


A dendrogram using Single Linkage strategy between two columns in this data set (Annual Income, Spending Score), This result is unclear and not obvious as we will see the scatter plot.

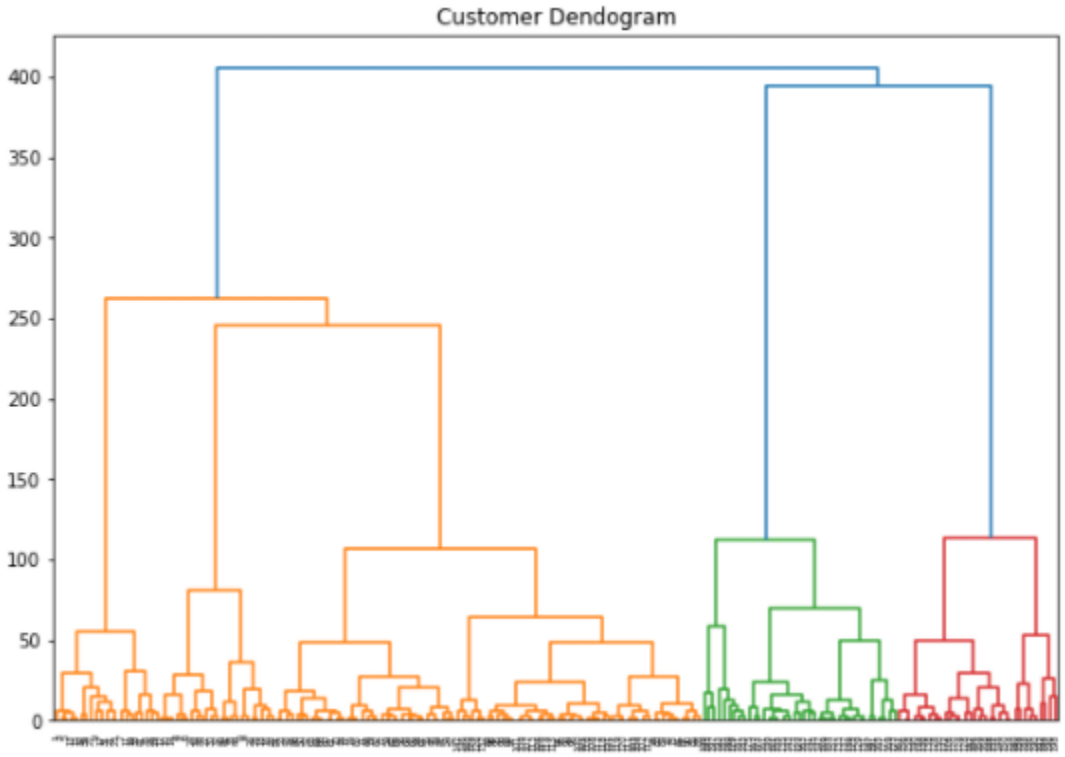
```
In [1044]: HierarchialCluster = AC(n_clusters =5,affinity ='euclidean',linkage ='single')
           HierarchialCluster.fit_predict(Final)

Out[1044]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 2, 4, 2,
                  3, 0], dtype=int64)
```

Agglomerative Hierarchal Clustering using 5 as the number of clusters, Euclidean as the measured distance & Single linkage strategy and this result is unsatisfying but it will be solved.



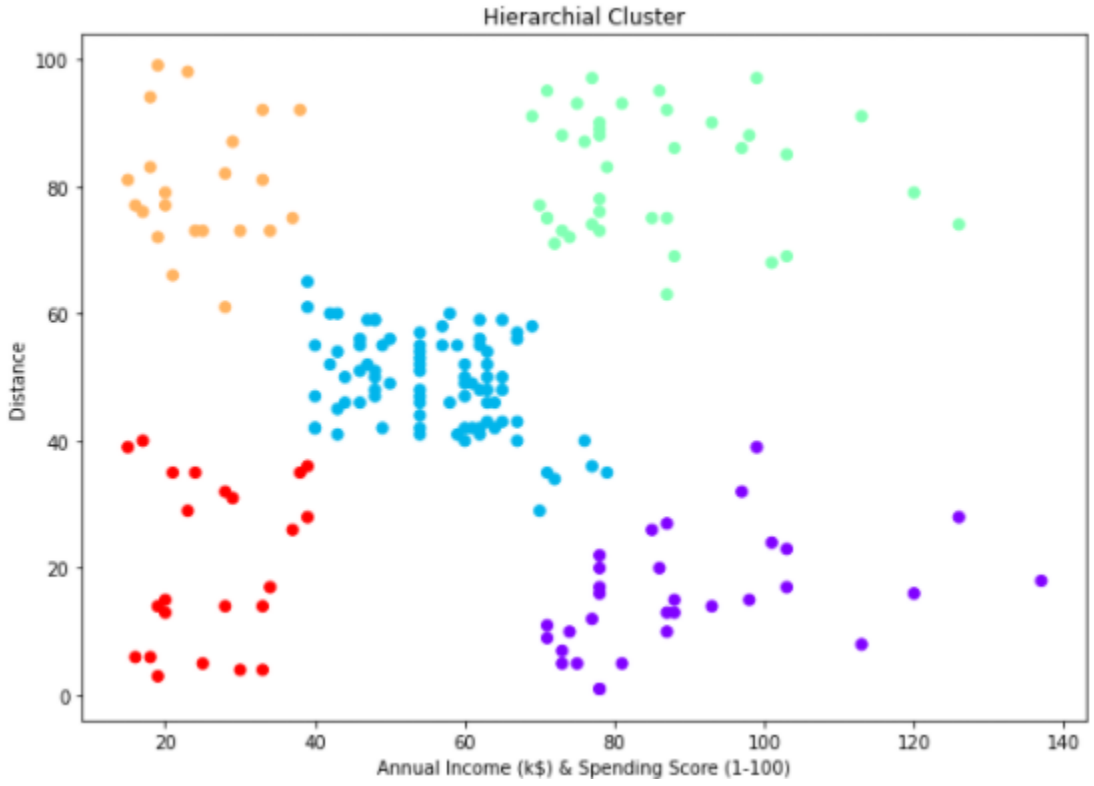
This scatter plot does not represent the desired result as single linkage.



Here is a much better result dendrogram using different linkage which is ward or complete and the number of clusters that are noticed is 5.

```
In [1047]: # Using ward linkage strategy to test the Hierarchial Clustering
           # Using ward linkage was excellent for Hierarchial Clustering
           HierarchialCluster1 = AC(n_clusters =5,affinity ='euclidean',linkage ='ward')
           HierarchialCluster1.fit_predict(Final)
```

```
Out[1047]: array([4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
                  4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 1,
                  4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 0, 2, 0, 2,
                  1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2,
                  0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
                  0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
                  0, 3], dtype=int64)
```



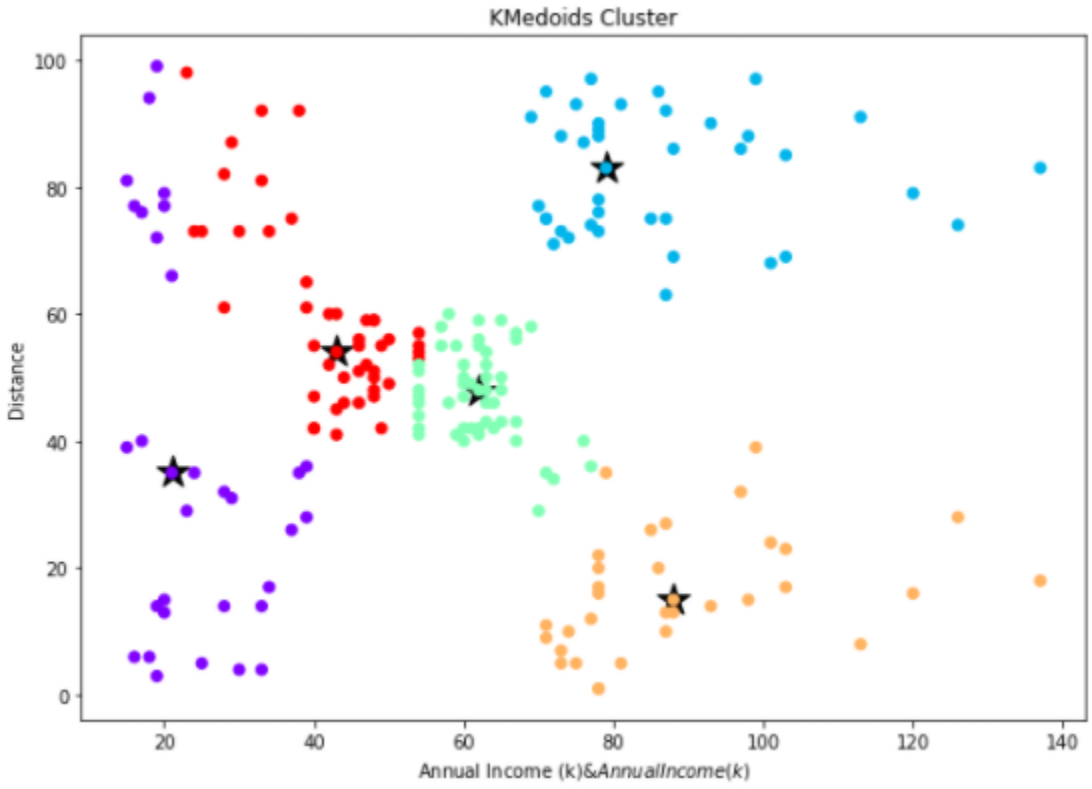
This really shows how powerful the hierarchal algorithm clustering methods.

1) 3 Part 2) Visualization: (K Medoids Clustering)
   Using 5 as the number of clusters, Manhattan as the measured distance.

```
In [1053]: # Secondly KMedoids WITH K=5 Clusters and Manhattan distance
           clu = KMedoids(n_clusters=5,metric="manhattan",init='random',random_state=33)
           clu.fit_predict(data)

Out[1053]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 4,
                  0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4,
                  0, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
                  4, 4, 4, 4, 4, 4, 4, 4, 2, 4, 4, 2, 2, 2, 2, 4, 2, 2, 4, 2, 2, 2,
                  2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                  2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 1, 2, 1, 3, 1, 3, 1,
                  2, 1, 3, 1, 3, 1, 3, 1, 3, 1, 2, 1, 3, 1, 2, 1, 3, 1, 3, 1, 3, 1,
                  3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
                  3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
                  3, 1], dtype=int64)
```

K medoids is definitely better here if we compare it with hierarchal clustering using single linkage only as it's the case here but hierarchal is better only if we use complete or ward linkage.



The (green and red) are not good result as well as (red and purple)

The black stars represent the five cluster centers which are an actual points of this dataset.

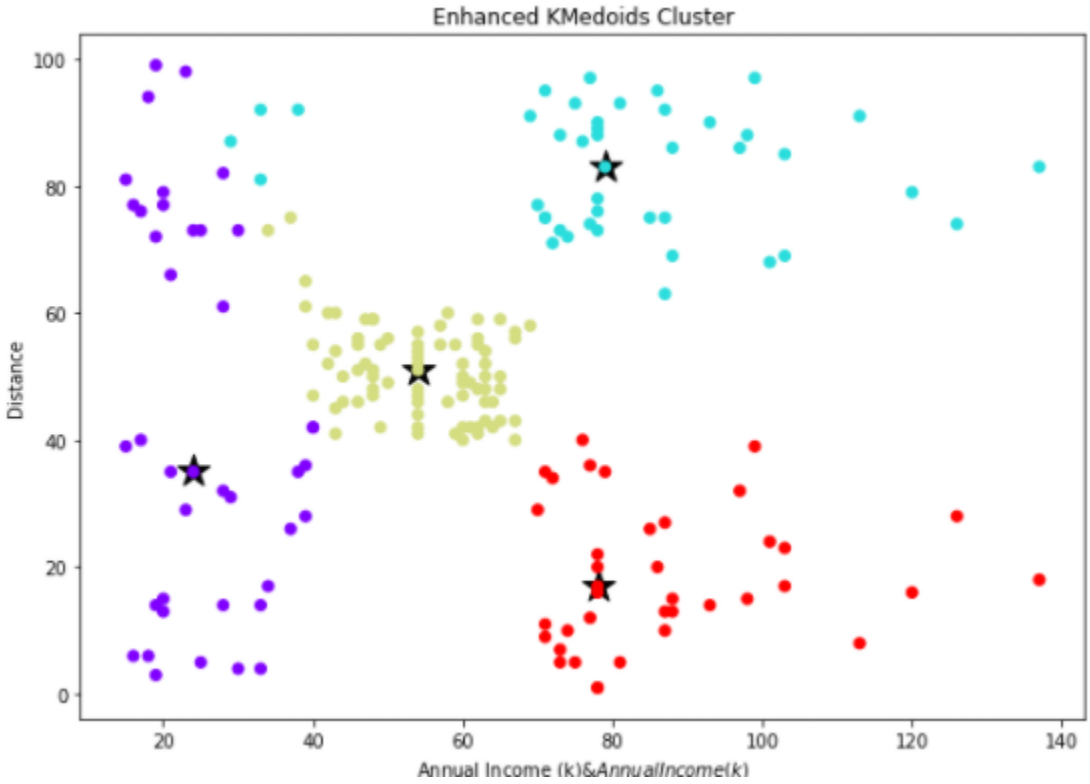
```
In [1056]: print('INDICES : ',clu.medoid_indices_)
           print('Cluster Centers : \n',clu.cluster_centers_)

INDICES :  [ 16 161 101 176  52]
Cluster Centers :
 [[21 35]
 [79 83]
 [62 48]
 [88 15]
 [43 54]]
```

Another K medoids with 4 cluster instead of 5 with Manhattan distance.

```
In [1061]: cluENH = KMedoids(n_clusters=4,metric="manhattan",init='random',random_state=33)
           cluENH.fit_predict(data)

Out[1061]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 2, 0, 2, 0, 1, 0, 2,
                  0, 2, 2, 2, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                  2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                  2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                  2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 3, 1, 3, 1, 3, 1, 3, 1,
                  3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
                  3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
                  3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
                  3, 1], dtype=int64)
```

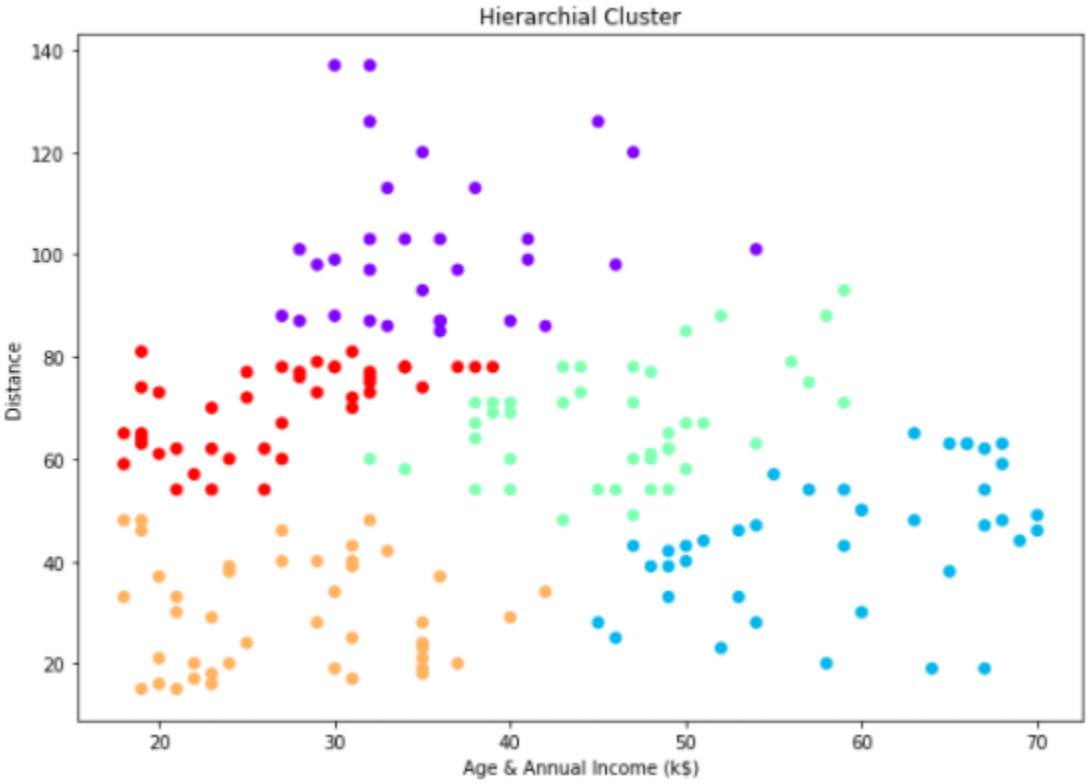


The previous picture has k medoids with 4 clusters but the result is more accurate but one less mistake as it‘s shown between (blue and purple) clusters so if the number of clusters is 4 then it ‘s better than using only 5 clusters in K medoids clustering.

Another Example

(Agglomerative Hierarchal Clustering)

Between columns (Age & Annual income) using 5 as the number of clusters and ward linkage again ass single linkage performance is weak here in this data.



This visualization is more accurate than the K medoid representation as it will be shown next as we focus on the green cluster exactly as the details here is better than k medoids cluster.

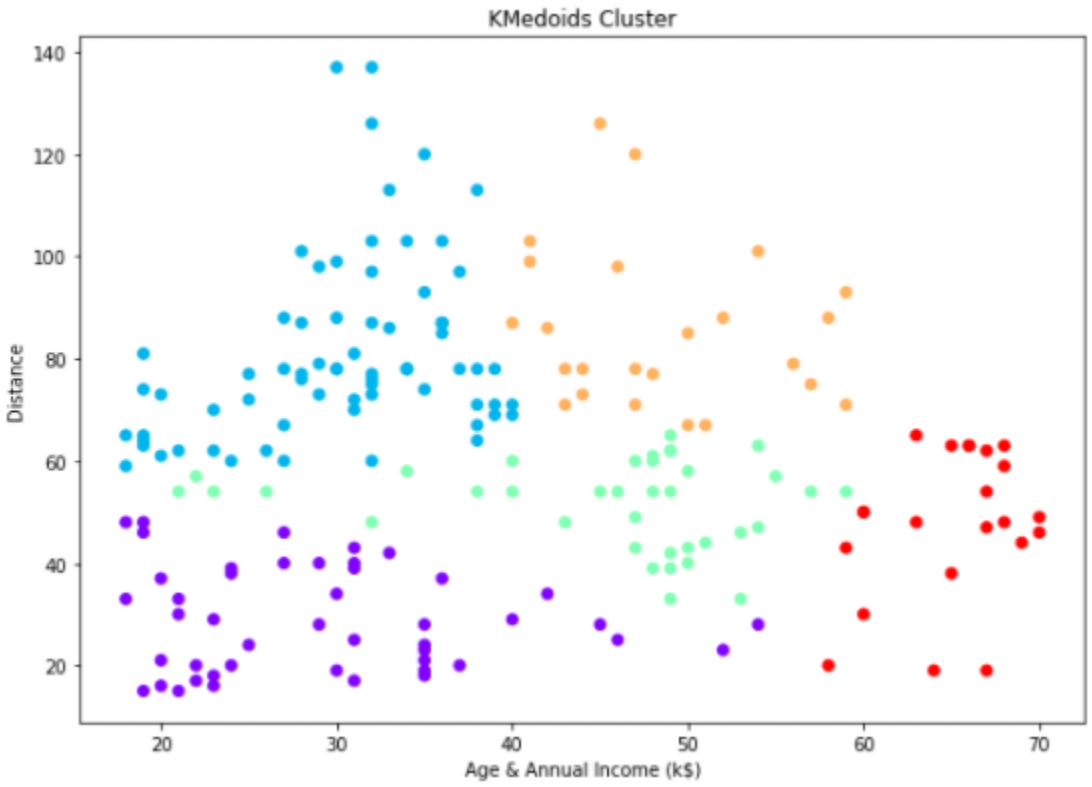
Cont: Another Example

(K Medoids Clustering)

Using 5 as the number of clusters, Manhattan as the measured distance

```
In [1057]: # Another Example Using Kmedoids
           clu1 = KMedoids(n_clusters=5,metric="manhattan",init='random',random_state=33)
           clu1.fit_predict(data1)
```

```
Out[1057]: array([0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 4, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 2, 0, 2, 0, 0, 0, 0, 0, 4, 0, 2, 0,
                  2, 0, 2, 0, 0, 0, 2, 0, 0, 4, 2, 2, 2, 4, 0, 2, 4, 0, 4, 2, 4, 0,
                  2, 4, 0, 2, 4, 2, 4, 4, 2, 2, 2, 2, 2, 2, 2, 2, 4, 2, 2, 2, 2, 2,
                  2, 2, 4, 1, 2, 2, 1, 1, 2, 1, 2, 1, 1, 2, 4, 1, 2, 1, 4, 2, 4, 4,
                  4, 1, 1, 1, 1, 1, 4, 2, 3, 3, 1, 1, 1, 1, 1, 1, 3, 1, 3, 1, 3, 1,
                  1, 1, 1, 1, 3, 1, 1, 1, 3, 1, 1, 1, 1, 1, 3, 1, 1, 1, 3, 1, 3, 1,
                  3, 1, 1, 1, 1, 1, 3, 1, 1, 1, 3, 1, 3, 1, 1, 1, 3, 1, 1, 1, 3, 1,
                  3, 1, 3, 1, 1, 1, 3, 1, 3, 1, 3, 1, 3, 1, 1, 1, 1, 1, 3, 1, 3, 1,
                  1, 1], dtype=int64)
```



As we focus on the green cluster we can notice that some data points do not belong to that green cluster but overall K medoids here has more accuracy than hierarchal if we had used the single linkage strat.

Comparison

Agglomerative Hierarchal Clustering	*K - Medoids Clustering*
Having dendrograms are an advantage as is shows how many clusters do you need in a visualization form is better.	You have to guess how many clusters at first then visualize your data to decide the appropriate number of clusters needed.
You have to decide which linkage work the best that give you the best result as single linkage performance was weak however Complete or ward linkage was more fitted in this situation so that was a disadvantage.	This clustering method works almost all time correctly which means a guaranteed good result almost every time so this is an advantage for K medoids as the result is accurate to a degree that makes a small difference only.
More Accurate result in specifying which point belong to which cluster exactly.	Overall a good less accurate result in specifying which point belong to which cluster as it was not very good at detecting that.
Both clusters algorithms require high computational power and as the data set gets more larger it becomes more difficult to apply hierarchal agglomerative clustering as K medoid works with more types of data but it requires more computing power so in the end every kind of clustering algorithm has it's purpose and in this project K Medoids is better definitely than Hierarchal as single linkage performance was bad	

Group: 1A

أحمد ياسر محمد محمد نصار

ID: 20191616316 Email: cds.AhmedYasser16316@alexu.edu.eg