Machine Learning In Class Exercise - 6
Hardik Bharatkumar Somaiya
Student ID- 700726838

Github Link:- https://github.com/hasomaiy/machine_learning_problems/blob/0e522643316798cd0beff8044553e0182b0cbfb3/Assignment6.ipynb

Video Link:- https://drive.google.com/drive/folders/1yF1lCGx0jOxCUmJmhbrQhMEo_29etQqk?usp=share_link

Question 1:-
We are being asked to calculate and find out clustering representations and dendrogram using Single, complete, and average link proximity function in hierarchical clustering technique.

Single:

We take minimum of points of clusters to be merged and continue merging till only 1 cluster reamains.

Cluster 1 is formed for pairs P3 and P6
Cluster 2 is formed for pairs P2 , P6
Cluster 3 is formed for pairs (P3,P6) and (P2,P5)
Final cluster is formed between P2,P3,P5,P6,P4

	P1	P2	P3	P4	P5	P6
P1	0.0					
P2	0.2357	0.0				
P3	0.2218	0.1483	0.0			
P4	0.3688	0.2042	0.1513	0.0		
P5	0.3421	0.1388	0.2843	0.2932	0.0	
P6	0.2347	0.2540	0.1100	0.2216	0.3921	0.0

In the distance matrix the minimum pair is

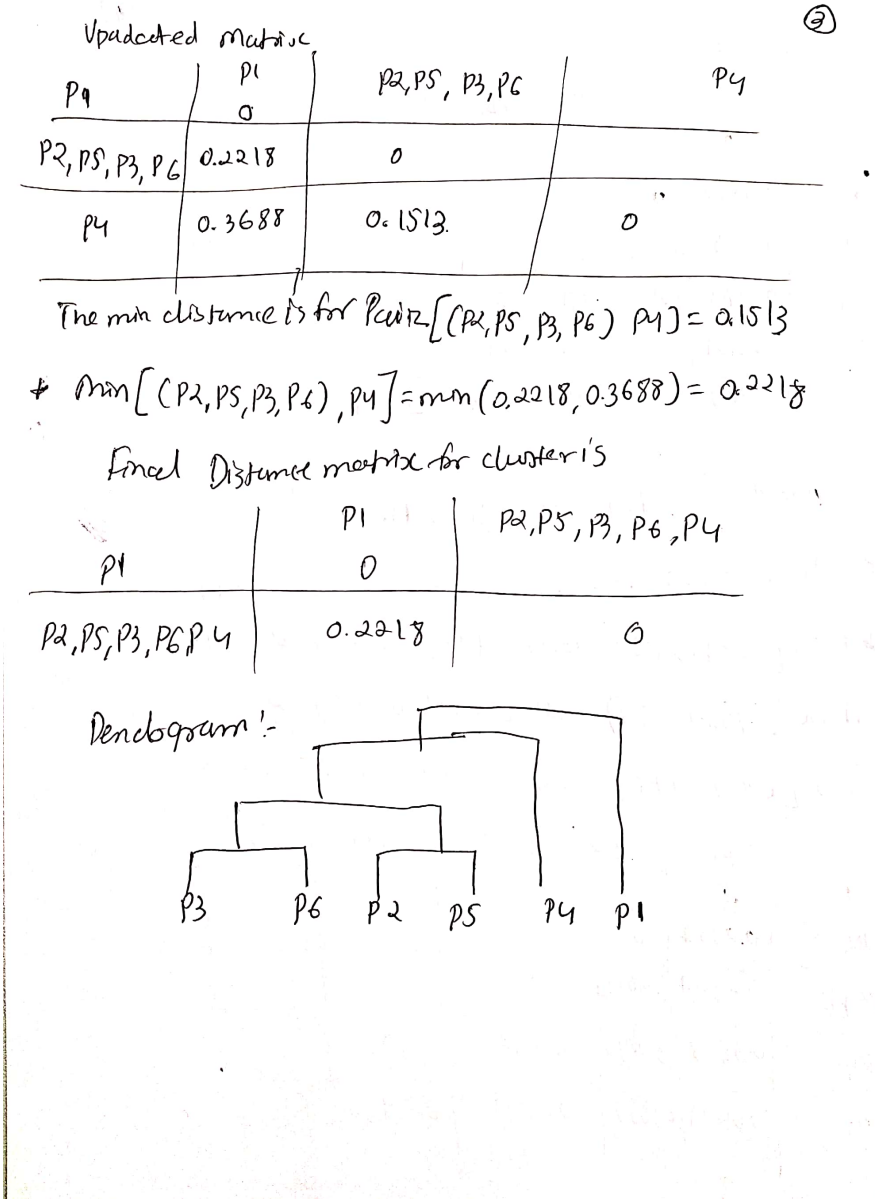
Pair [P3, P6] = 0.11

Updating the distance matrix $\min[dist(P3,P6), P1]$

$\min(dist(P3,P1),(P6,P1)) = \min(0.2218, 0.2347) = 0.2218$

$\min(dist(P3,P6)P2) = \min(dist(P3,P2), dist(P6,P2)) = \min(0.1483, 0.2540) = 0.1483$

Final Matrix and Dendogram is as follows:



Updated matrix

	P1	P2, P5, P3, P6	P4
P1	0		
P2, P5, P3, P6	0.2218	0	
P4	0.3688	0.1513	0

The min distance is for P(1,2,5,3,6) P4] = 0.1513

* min [(P2, P5, P3, P6), P4] = min (0.2218, 0.3688) = 0.2218

Final Distance matrix for cluster's

	P1	P2, P5, P3, P6, P4
P1	0	
P2, P5, P3, P6 P4	0.2218	0

Dendogram :-

Complete Link:-

First we select minimum points and start building clusters
We take maximum of two minimum points and create clusters depending on total clusters we need. Here the clusters are formed between P3 and P6 then P2,P5 and P4 & (P3,P5).
Then between P1 and (P2,P5) .

Final cluster is (P1,P2,P5) and (P3,P6,P4)

(4)

(b) Complete Link.

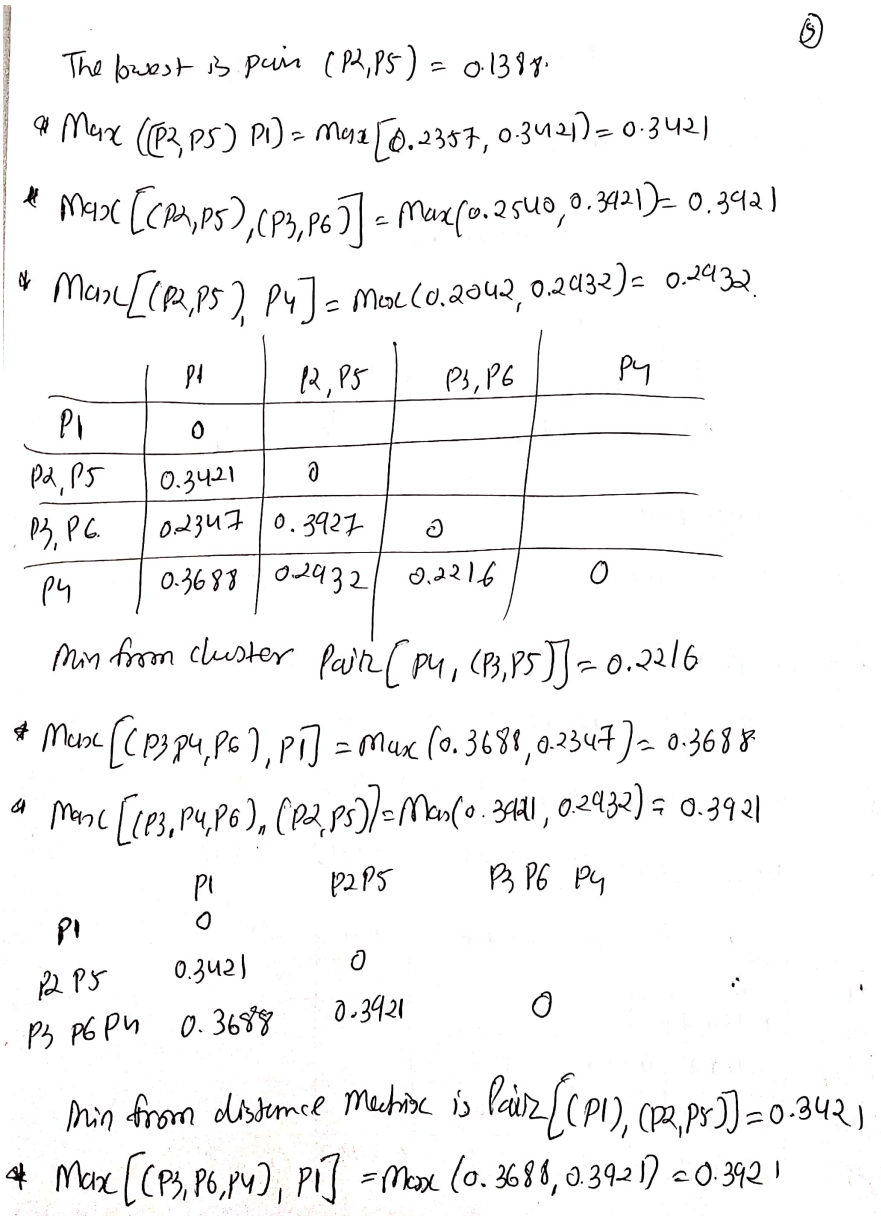
	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.2357	0				
P3	0.2218	0.1483	0			
P4	0.3688	0.2042	0.1513	0		
P5	0.3421	0.1387	0.2843	0.2433	0	
P6	0.2347	0.2540	0.1100	0.2216	0.3922	0

Min distance Pair (P3, P6) = 0.1100

* Max [(P3,P6), P1] = Max (0.2218, 0.2347) = 0.2347
* Max [(P3,P6) P2] = Max (0.1483, 0.2540) = 0.2540
* Max [(P3,P6) P4] = Max (0.1513, 0.2216) = 0.2216
* Max [(P3,P6), P5] = Max (0.2843, 0.3922) = 0.3922

	P1	P2	P3,P6,	P4	P5
P1	0				
P2	0.2357	0			
P3,P6	0.2347	0.2540	0		
P4	0.3688	0.2042	0.2216	0	
P5	0.3421	0.1387	0.3421	0.2932	0

Dendogram:



⑤

The lowest is pair $(P2, P5) = 0.1398$.

* $\text{Max}((P2, P5), P1) = \text{Max}[0.2357, 0.3421) = 0.3421$
* $\text{Max}([(P2, P5), (P3, P6)]) = \text{Max}(0.2540, 0.3421) = 0.3921$
* $\text{Max}[(P2, P5), P4] = \text{Max}(0.2042, 0.2432) = 0.2432$

	P1	P2, P5	P3, P6	P4
P1	0			
P2, P5	0.3421	0		
P3, P6	0.2347	0.3927	0	
P4	0.3688	0.2432	0.2216	0

Min from cluster Pair$[P4, (P3, P5)] = 0.2216$

* $\text{Max}[(P3, P4, P6), P1] = \text{Max}(0.3688, 0.2347) = 0.3688$
* $\text{Max}[(P3, P4, P6), (P2, P5)] = \text{Max}(0.3421, 0.2432) = 0.3921$

	P1	P2 P5	P3 P6 P4
P1	0		
P2 P5	0.3421	0	
P3 P6 P4	0.3688	0.3921	0

Min from distance Matrix is Pair$[(P1), (P2, P5)] = 0.3421$

* $\text{Max}[(P3, P6, P4), P1] = \text{Max}(0.3688, 0.3921) = 0.3921$

Average Link

The average link takes the average between the distances of two minimum data points to form clusters.

the clusters are formed as : P3 and P6,

between P2 and P5, then, (P3,P6) and P4.

between (P5,P2) and (P3,P4,P6).

Dendogram:

Question 2:-

a. preprocessing by removing null values and replacing it by mean of the column and then dropping the categorical column

+ Avg[(P2,P3,P4,P5,P6),P1] = Avg(0.2889, 0.2985) = 0.2937

c) Now, I have used PCA and reduced the attributes to 2 dimensions.

(c) ¶

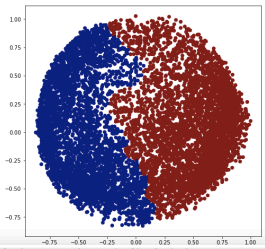
```
In [114]: pca2 = PCA(n_components=2)
          principalComponents = pca2.fit_transform(X_Norm)        #Applying PCA and reducing the no. features to 2

          principalDf = pd.DataFrame(data = principalComponents, columns = ['principal component 1', 'principal component 2'])

          principalDf.head()
```

Out[114]:

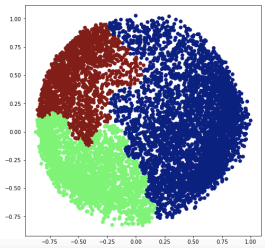
	principal component 1	principal component 2
0	-0.489826	-0.679678
1	-0.518792	0.545011
2	0.330885	0.268978
3	-0.482374	-0.092111
4	-0.563289	-0.481915

d) After applying PCA, I have now used scatter plot to visualize the agglomerative clustering using by taking k(no. of clusters) as 2,3,4,5

```
In [115]: # Using scatter plot to visualize for k =2
          Agglo_Cluster2 = AgglomerativeClustering(n_clusters = 2)
          plt.figure(figsize =(8,8))
          plt.scatter(principalDf['principal component 1'], principalDf['principal component 2'],
          c = Agglo_Cluster2.fit_predict(principalDf), cmap ='jet')
          plt.show()
```

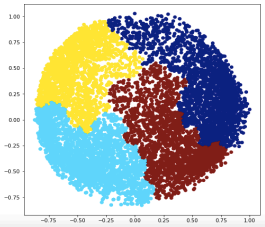



```
In [116]: # Using scatter plot to visualize for k =3
          Agglo_Cluster3 = AgglomerativeClustering(n_clusters = 3)
          plt.figure(figsize =(8,8))
          plt.scatter(principalDf['principal component 1'], principalDf['principal component 2'],
          c = Agglo_Cluster3.fit_predict(principalDf), cmap ='jet')
          plt.show()
```

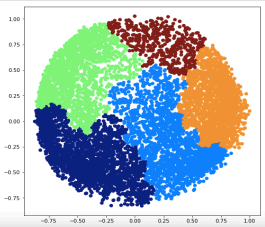


+ Avg[(P2,P5),(P3,P6)] = Avg(0.2011, 0.3382) = 0.2446

```
In [18]: # Using scatter plot to visualize for k = 4
Agglo_Cluster4 = AgglomerativeClustering(n_clusters = 4)
plt.figure(figsize =(8,8))
plt.scatter(principalDf['principal component 1'], principalDf['principal component 2'],
c = Agglo_Cluster4.fit_predict(principalDf), cmap ='jet')
plt.show()
```



```
In [118]: # Using scatter plot to visualize for k = 5
Agglo_Cluster5 = AgglomerativeClustering(n_clusters = 5)
plt.figure(figsize =(8,8))
plt.scatter(principalDf['principal component 1'], principalDf['principal component 2'],
c = Agglo_Cluster5.fit_predict(principalDf), cmap ='jet')
plt.show()
```

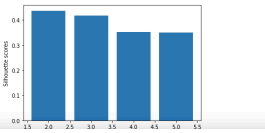


e) Finally, I have calculated the silhouette score for each clusters and them compared them using a bar plot.

```
print("Silhouetter score for 2 clusters is =", Silhouette[0])
Silhouette.append(silhouette_score(principalDf, Agglo_Cluster3.fit_predict(principalDf)))
print("Silhouetter score for 3 clusters is =", Silhouette[1])
Silhouette.append(silhouette_score(principalDf, Agglo_Cluster4.fit_predict(principalDf)))
print("Silhouetter score for 4 clusters is =", Silhouette[2])
Silhouette.append(silhouette_score(principalDf, Agglo_Cluster5.fit_predict(principalDf)))
print("Silhouetter score for 5 clusters is =", Silhouette[3])
```

```
Silhouetter score for 2 clusters is = 0.4373242834730189
Silhouetter score for 3 clusters is = 0.4167229587664966
Silhouetter score for 4 clusters is = 0.3531061921312927 4
Silhouetter score for 5 clusters is = 0.3505598240817038 5
```

```
In [21]: # Plotting a bar graph to compare the results
plt.bar(k, Silhouette)
plt.xlabel('Number of clusters', fontsize = 10)
plt.ylabel('Silhouette scores', fontsize = 10)
plt.show()
```



Here I can conclude that the best silhouette score is when I use 2 cluster which is 43.7324%.