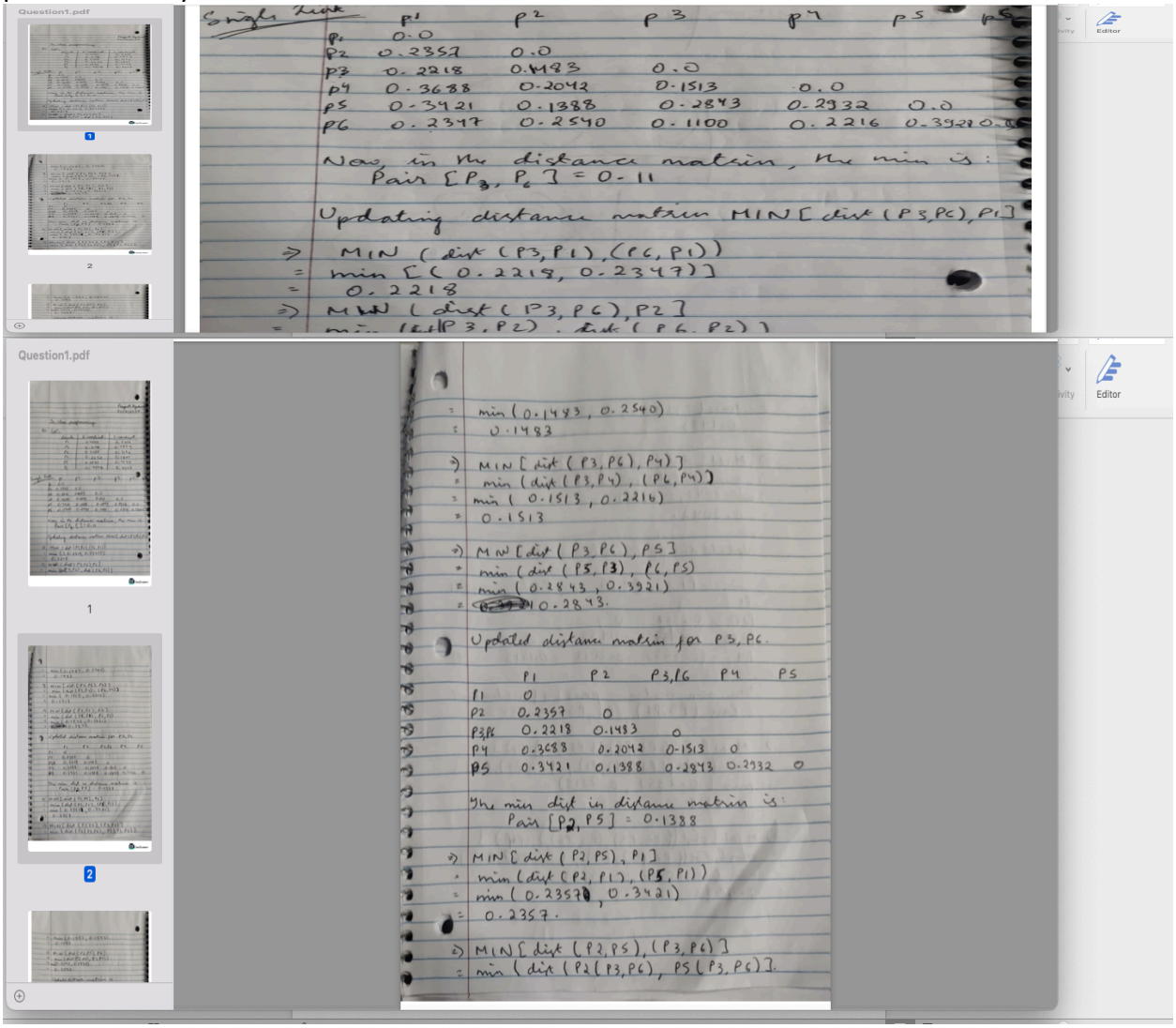
Assignment-6

Github Link : https://github.com/AayushSanghai/Machine-Learnin/tree/main/Assignment-6

Question 1:

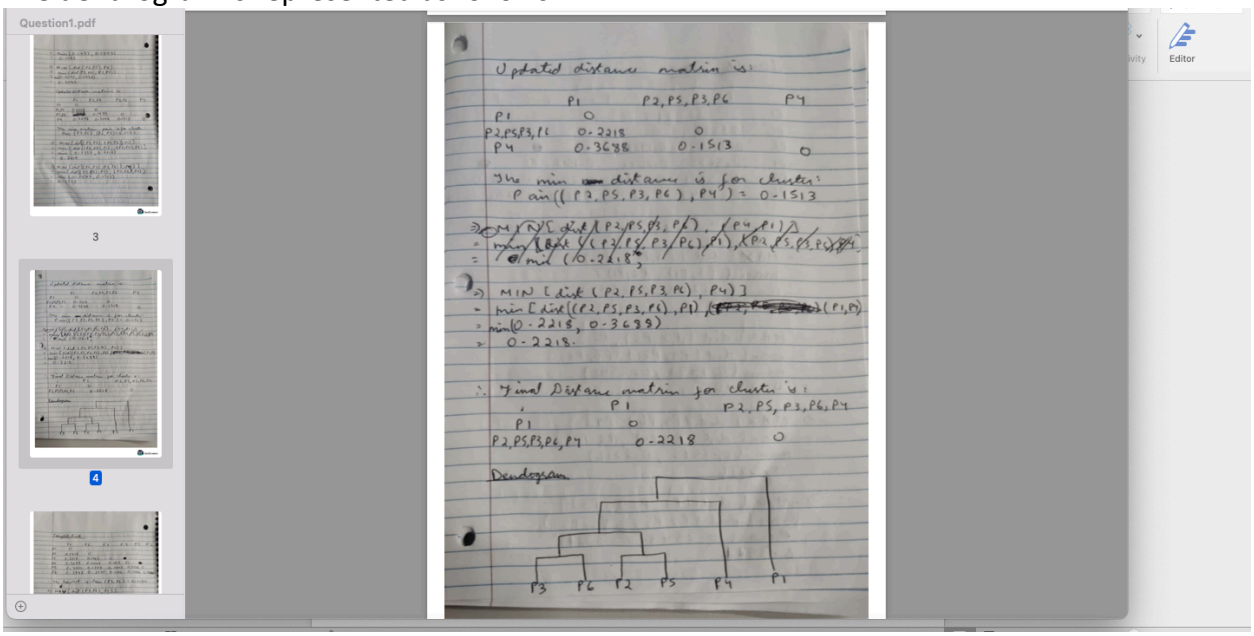
The x coordinate and y coordinate is given in the question. For our ease, the distance matrix is also provided which makes calculations less.
We are asked to calculate and the clustering representation for Hierarchical clustering and also draw dendrograms for Single, Complete and Average links.

Single Link:
We take the minimum of the points of the clusters that has to be merged and continue the process till only 1 cluster remains. Below are the screenshots”



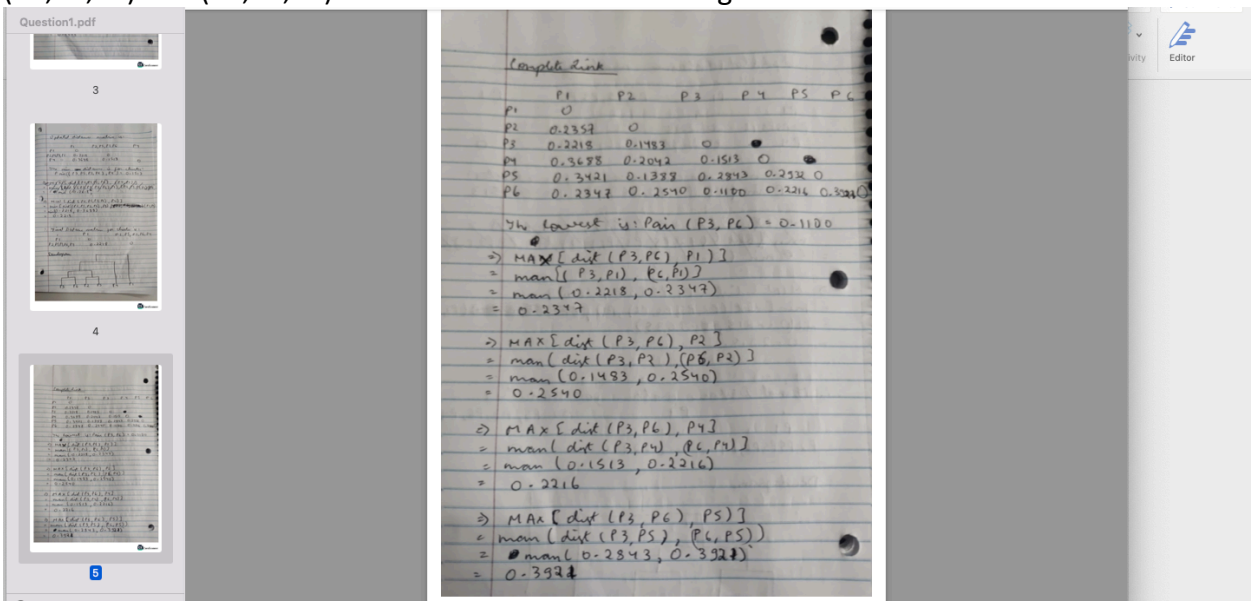
Here, 1st cluster is formed for pairs P3, P6, 2nd cluster is formed for P2, P5, 3rd cluster is formed between (P3,P6) and (P2,P5) and finally the final cluster is formed between P2,P3,P5,P5 and P4.

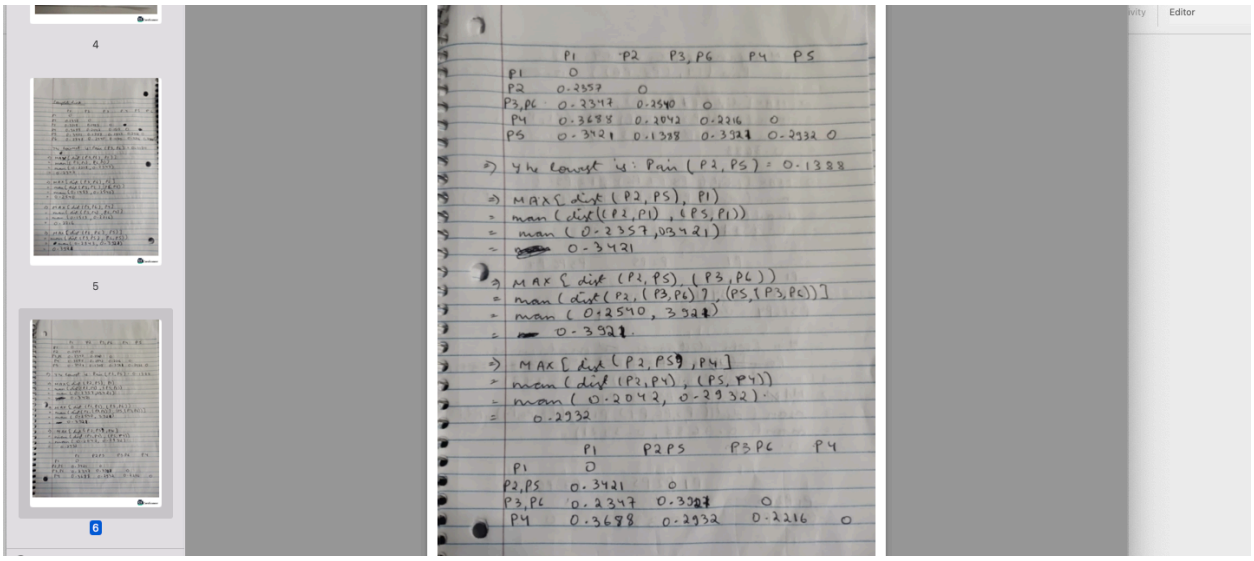
The dendrogram is represented as follows:

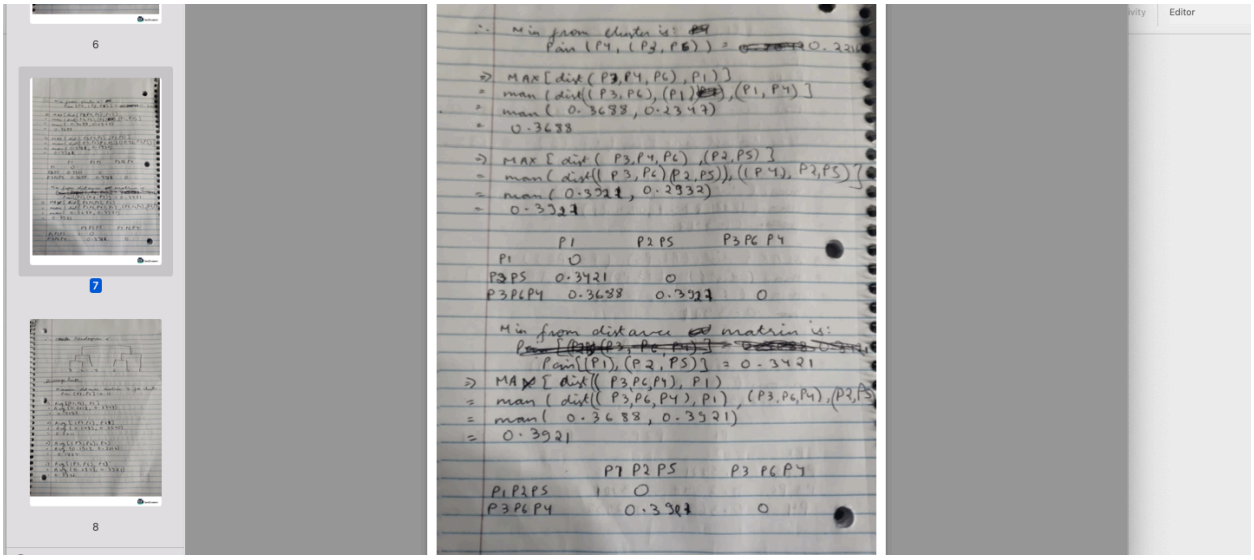


Complete Link:

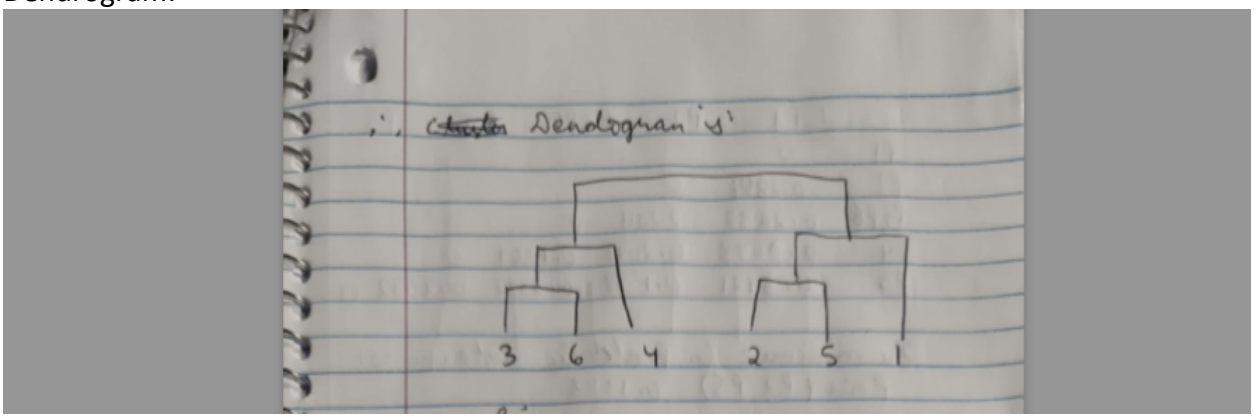
Here, we select the minimum point and then begin creating clusters. We take the maximum of the two minimum data points distances and then start creating clusters depending on the total no. of clusters we need. Here, the clusters formed for this assignment are P3 and P6, then P2,P5, then between P4 and (P3 and P5). Then between P1 and (P2,P5). Finally it ends with (P1,P2,P5) and (P3,P6,P4). The screenshots and dendrogram is attached below.





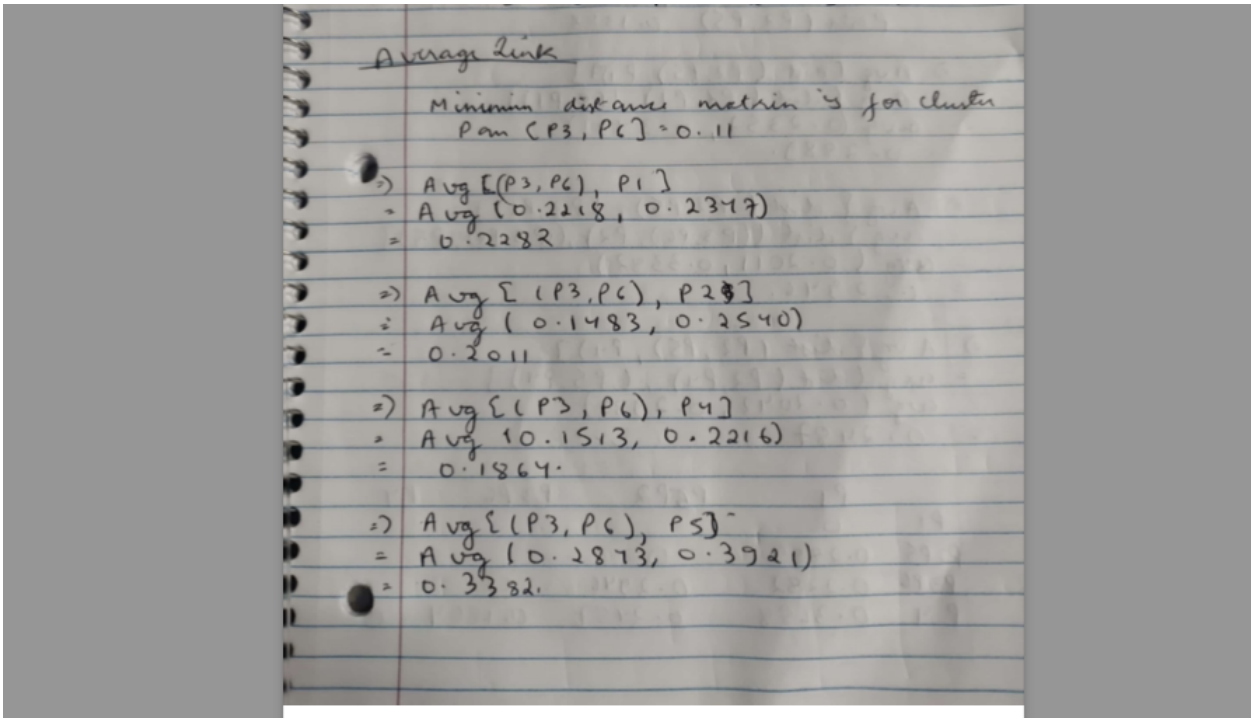


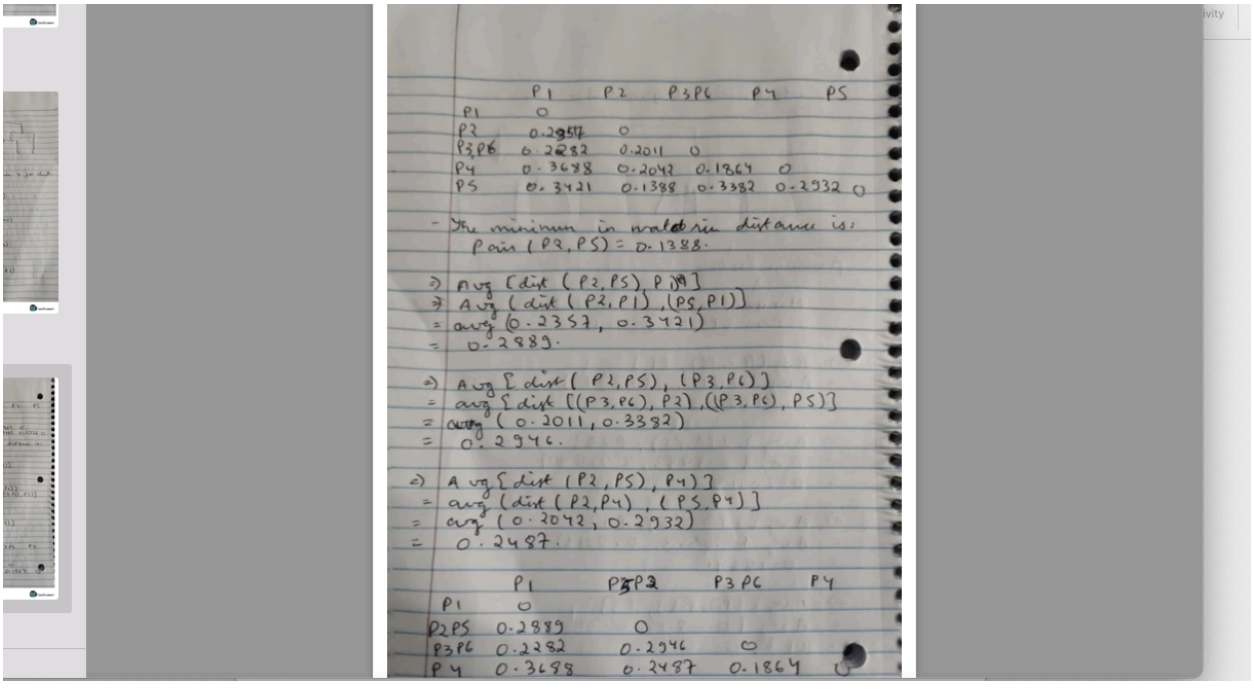
Dendrogram:

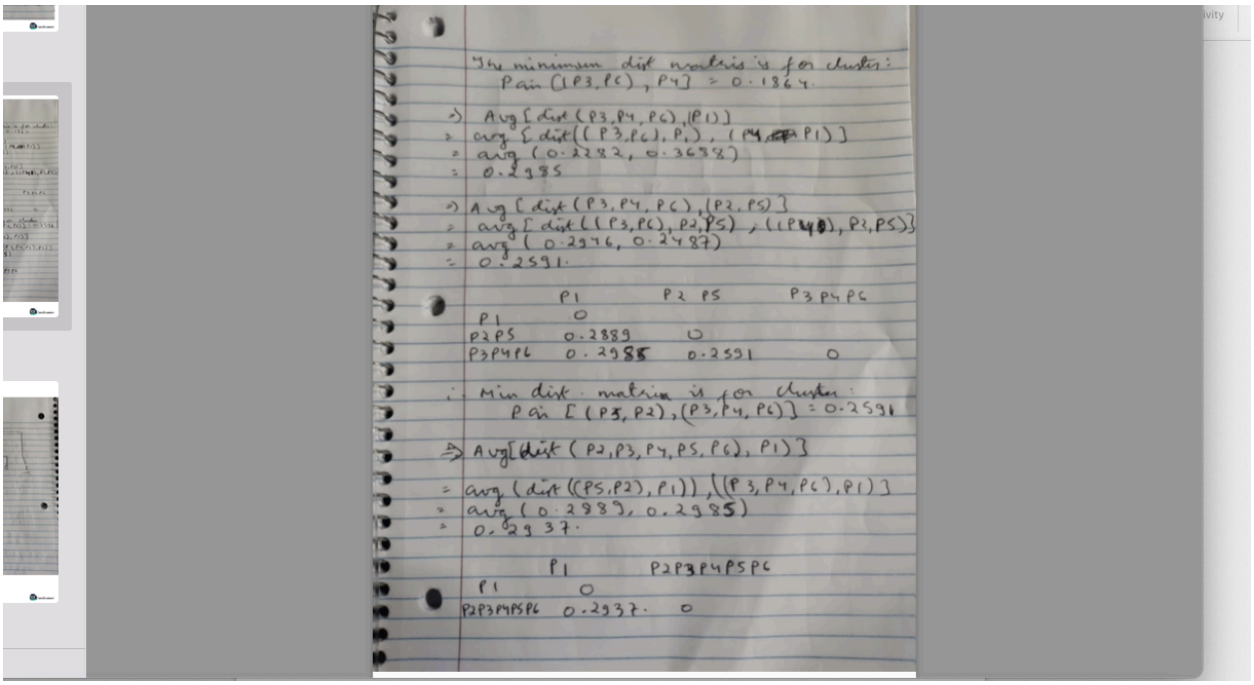


Average Link:

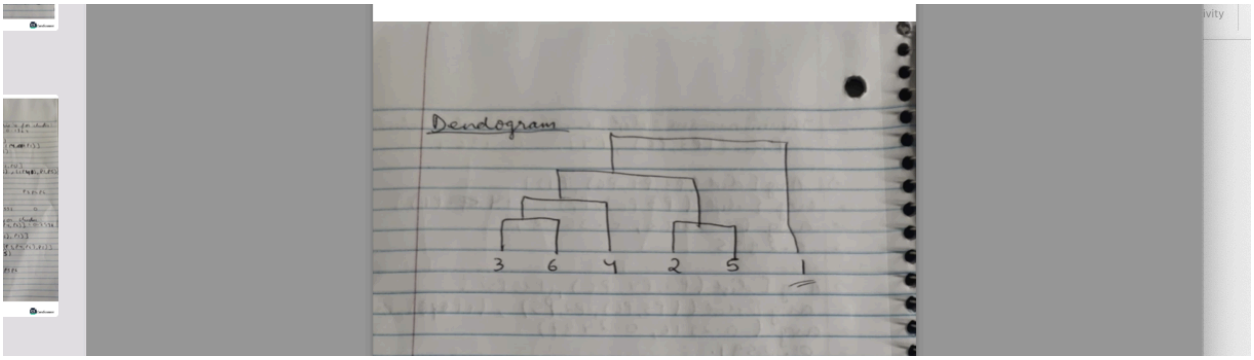
Unlike single link and complete link, the average link takes the average between the distances of two minimum data points to form clusters. For this assignment, the clusters are formed as : P3 and P6, then between P2 and P5, then, (P3,P6) and P4. Then between (P5,P2) and (P3,P4,P6). Below are screenshots of cluster formation and the distance matrix:







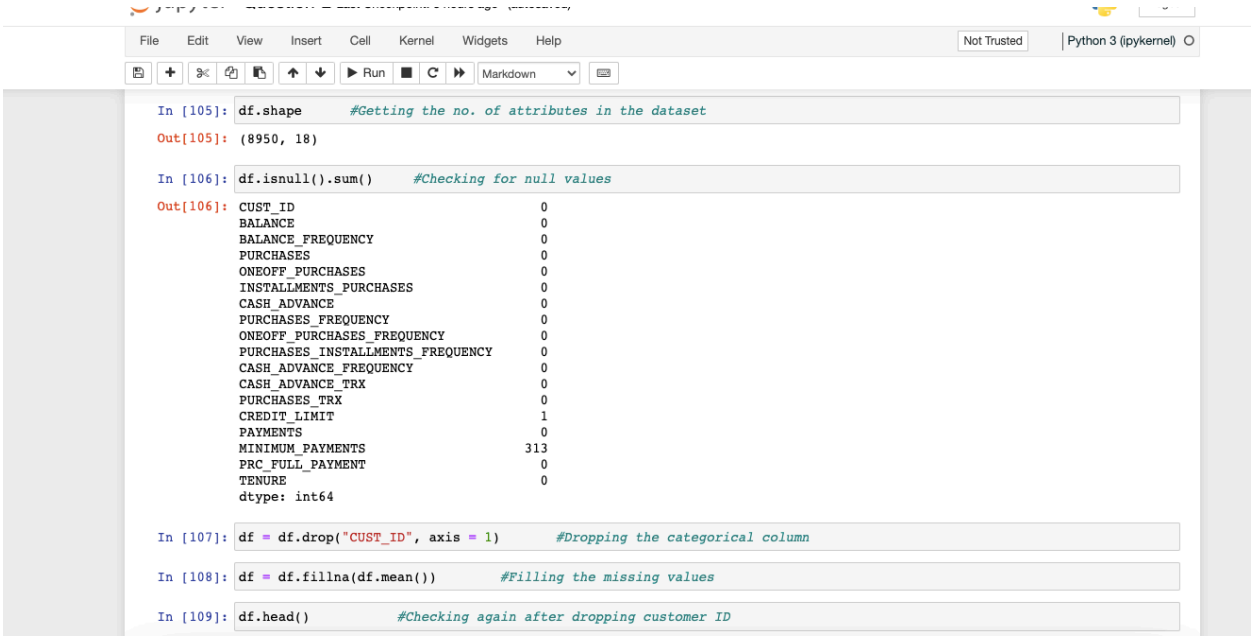
Dendrogram:



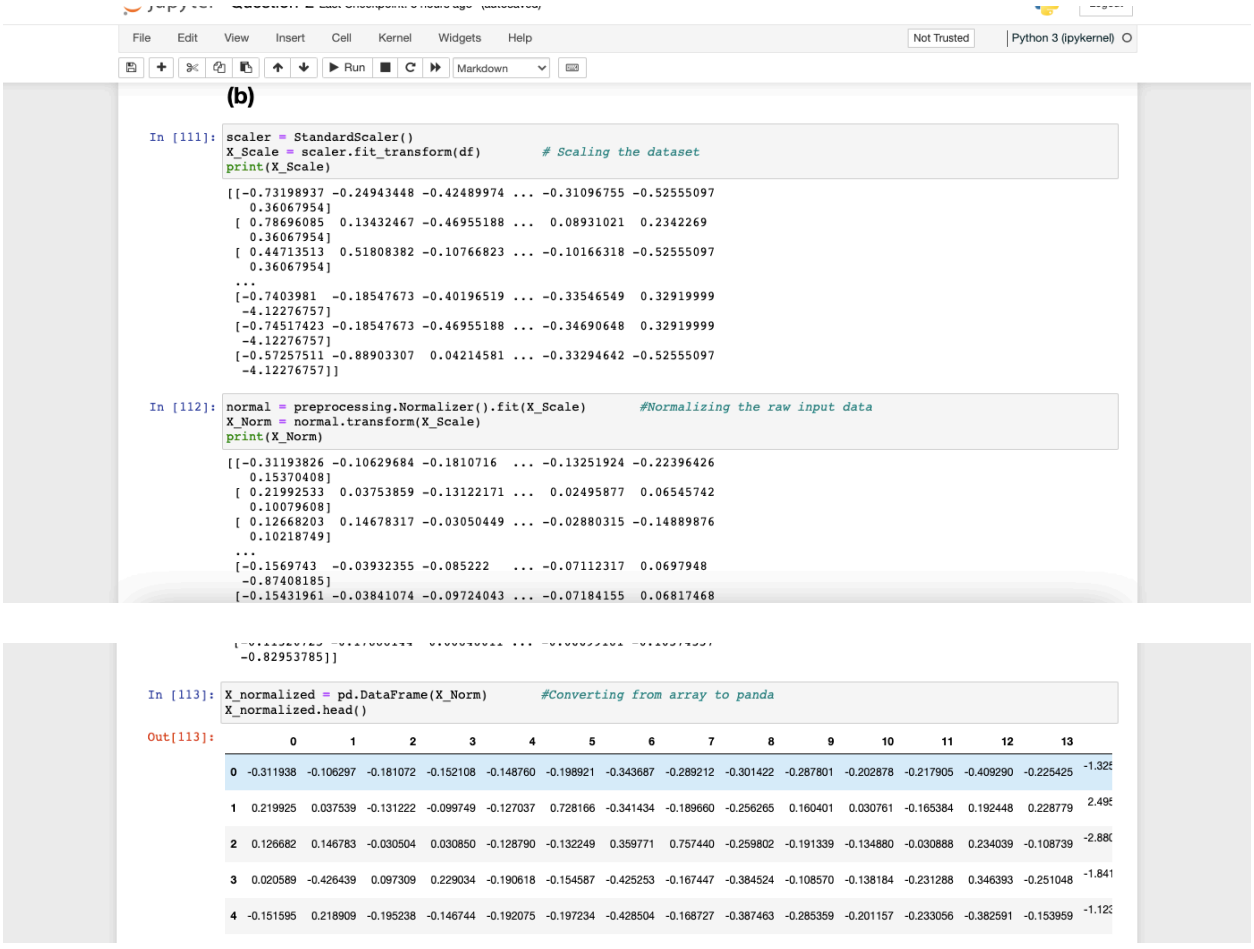
Question-2

For question 2, we are supposed to analyze the dataset CC General and then perform basic data preprocessing, then normalize and scale the data and the apply agglomerative clustering with different no. of clusters and finally calculation the silhouette score to analyze the accuracy of the mode.

a) Firstly, I have done the preprocessing by removing null values and replacing it by mean of the column and then dropping the categorical column



b) Then, I have scaled the data using Standard Scaler and then normalized the raw data points using the normalize() function



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

(c) ¶

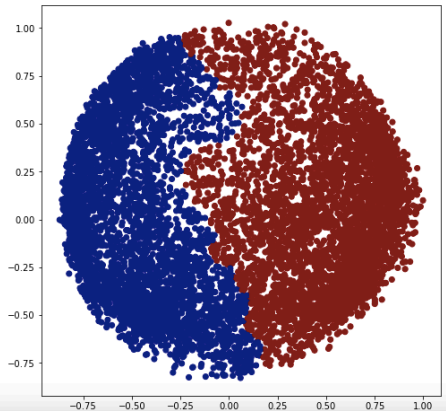
In [114]:
```
pca2 = PCA(n_components=2)
principalComponents = pca2.fit_transform(X_Norm)       #Applying PCA and reducing the no. features to 2

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

principalDf.head()
```

Out[114]:

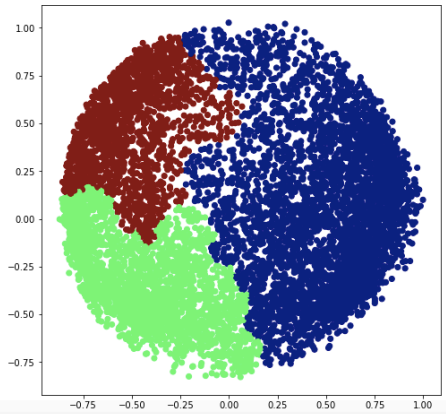
	principal component 1	principal component 2
0	-0.489826	-0.679678
1	-0.518792	0.545011
2	0.330885	0.268978
3	-0.482374	-0.092111
4	-0.563289	-0.481915

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

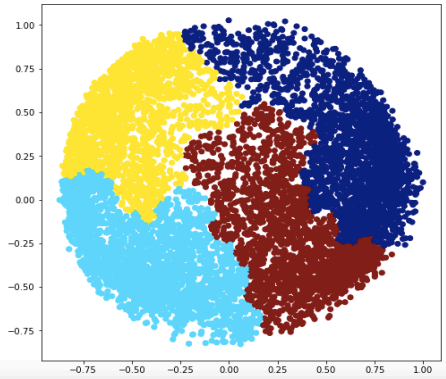
In [115]:
```
# Using scatter plot to visualize for k =2
Agglo_Cluster2 = AgglomerativeClustering(n_clusters = 2)
plt.figure(figsize =(8,8))
plt.scatter(principalDf['principal component 1'], principalDf['principal component 2'],
c = Agglo_Cluster2.fit_predict(principalDf), cmap ='jet')
plt.show()
```



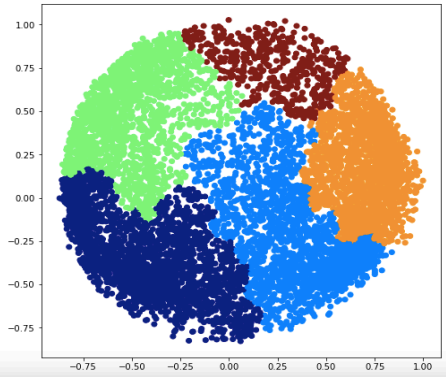
In [116]:
```
# Using scatter plot to visualize for k =3
Agglo_Cluster3 = AgglomerativeClustering(n_clusters = 3)
plt.figure(figsize =(8,8))
plt.scatter(principalDf['principal component 1'], principalDf['principal component 2'],
c = Agglo_Cluster3.fit_predict(principalDf), cmap ='jet')
plt.show()
```



```
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()
```

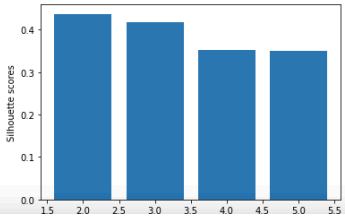


e) Finally, I have calculated the silhouette score for each clusters and them compared them using a bar plot.

```
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 = 0.35310619213129274
Silhouetter score for 5 clusters is = 0.35055082408170385
```

```
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%.