Machine Learning
Assignment 6

Name: Yeshwanth Reddy Kunam
Id: 700731889

Github link: https://github.com/yxk18890/ML_Assignment_6

1. Mathematical Solution:

Single Link Proximity:

- In **Single Linkage,** the distance between two clusters is the minimum distance between members of the two clusters

	p1	**p2**	**p3**	**p4**	**p5**	**p6**
p1	0	0.2357	0.2218	0.3688	0.3421	0.2347
p2	0.2357	0	0.1483	0.2042	0.1388	0.254
p3	0.2218	0.1483	0	0.1513	0.2843	0.11
p4	0.3688	0.2042	0.1513	0	0.2932	0.2216
p5	0.3421	0.1388	0.2843	0.2932	0	0.3921
p6	0.2347	0.254	0.11	0.2216	0.3921	0

Smallest distance from above data is 0.11

Hence, p3 and p6 forms first cluster.

	p1	p2	p3	p4	p5
p1	0	0.2357	0.2218	0.3688	0.3421
p2	0.2357	0	0.1483	0.2042	0.1388
p36	0.2218	0.1483	0	0.1513	0.2843
p4	0.3688	0.2042	0.1513	0	0.2932
p5	0.3421	0.1388	0.2843	0.2932	0

Smallest distance from above data is 0.1388

Therefore, p2 and p5 form second cluster.

	p1	p25	p36	p4
p1	0	0.2357	0.2218	0.3688
p2	0.2357	0	0.1483	0.2042
p3p6	0.2218	0.1483	0	0.1513
p4	0.3688	0.2042	0.1513	0

So, smallest distance from above is 0.1483.

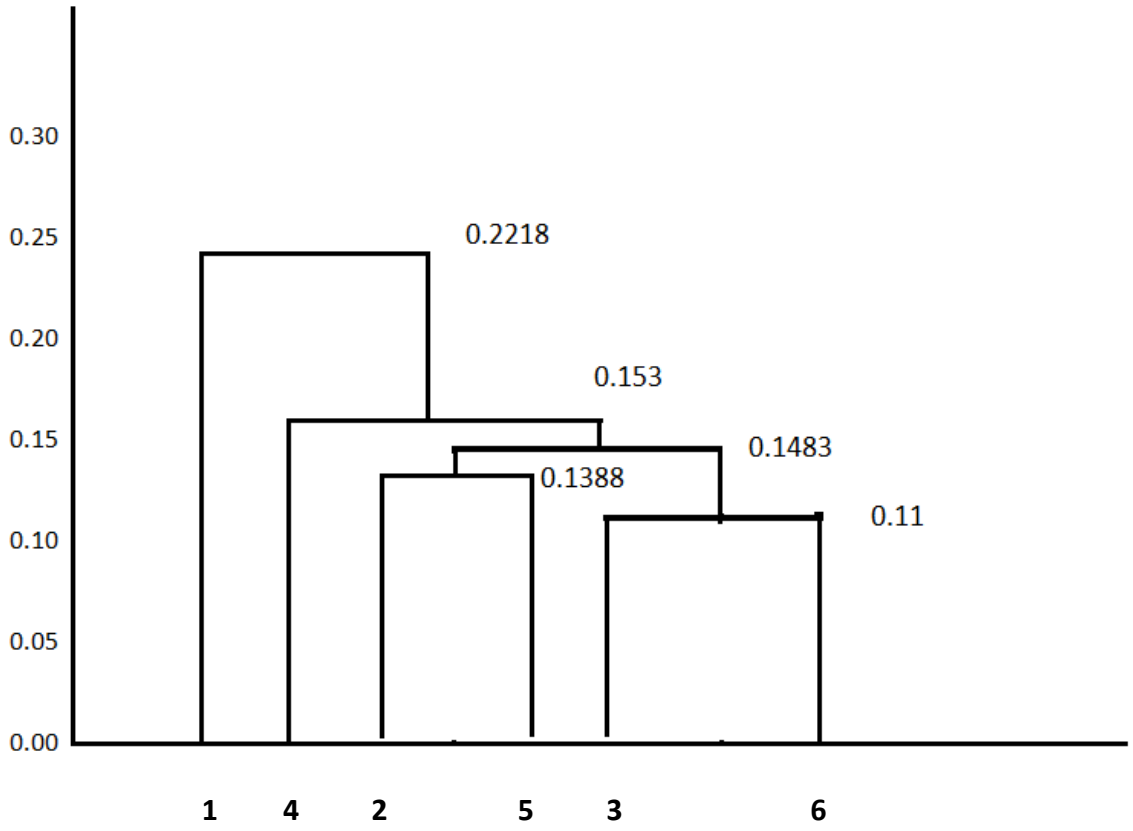
So, p25 and p36 forms third cluster.

	p1	p25(36)	p4
p1	0	0.2218	0.3688
p25p36	0.2218	0	0.1513
p4	0.3688	0.1513	0

Smallest distance from above is 0.1513

Hence, p(25)(36) and p4 forms fourth cluster.

	p1	p4(25)(36)
p1	0	0.2218
p4(25)(36)	0.2218	0



Complete link Proximity:

In **Complete Linkage,** the distance between two clusters is the maximum distance between members of the two clusters.

	p1	p2	p3	p4	p5	p6
p1	0	0.2357	0.2218	0.3688	0.3421	0.2347
p2	0.2357	0	0.1483	0.2042	0.1388	0.254
p3	0.2218	0.1483	0	0.1513	0.2843	0.11
p4	0.3688	0.2042	0.1513	0	0.2932	0.2216
p5	0.3421	0.1388	0.2843	0.2932	0	0.3921
p6	0.2347	0.254	0.11	0.2216	0.3921	0

Smallest distance from above data is 0.11
Hence, p3 and p6 forms first cluster.

	p1	p2	p36	p4	p5
p1	0	0.2357	0.2347	0.3688	0.3421
p2	0.2357	0	0.254	0.2042	0.1388
p36	0.2347	0.254	0	0.2216	0.3921
p4	0.3688	0.2042	0.2216	0	0.2932

p5	0.3421	0.1388	0.3921	0.2932	0

Smallest distance from above is 0.1388

Therefore, p2 and p5 form second cluster.

	p1	**p25**	**p36**	**p4**
p1	0	0.3421	0.2347	0.3688
p25	0.3421	0	0.3921	0.2932
p36	0.2347	0.3921	0	0.2216
p4	0.3688	0.2932	0.2216	0

Smallest distance from above is 0.2216

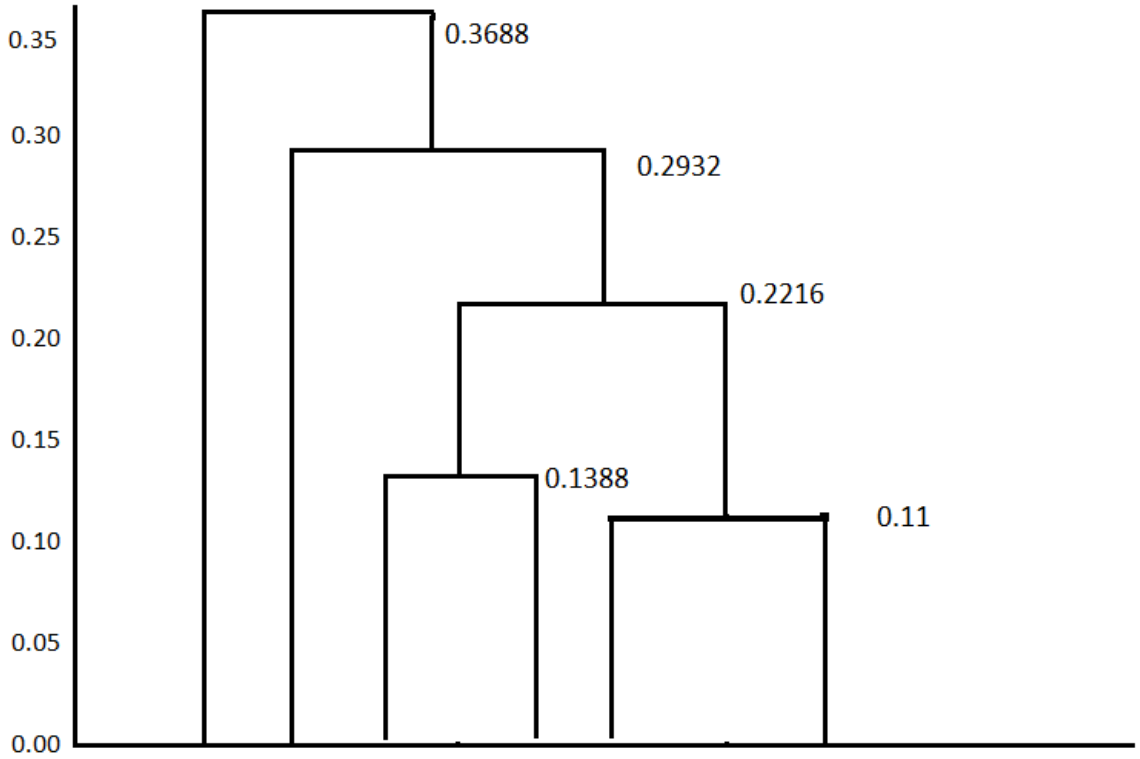
Hence, p36 and p25 form third cluster.

	p1	**p(25)(36)**	**p4**
p1	0	0.3421	0.3688
p(25)(36)	0.3421	0	0.2932
p4	0.3688	0.2932	0

Smallest distance from above is 0.2932

Hence p(25)(36) and p1 form fourth cluster.

	p1(25)(36)	**p4**
p1(25)(36)	0	0.1483
p4	0.3688	0



4 1 2 5 3 6

Average Link Proximity:

In **Average Linkage,** the distance between two clusters is the average of all distances between members of the two clusters

	p1	p2	p3	p4	p5	p6
p1	0	0.2357	0.2218	0.3688	0.3421	0.2347
p2	0.2357	0	0.1483	0.2042	0.1388	0.254
p3	0.2218	0.1483	0	0.1513	0.2843	0.11
p4	0.3688	0.2042	0.1513	0	0.2932	0.2216
p5	0.3421	0.1388	0.2843	0.2932	0	0.3921
p6	0.2347	0.254	0.11	0.2216	0.3921	0

Smallest distance from above is 0.11

Hence, p3 and p6 forms first cluster.

	p1	p2	p36	p4	p5
p1	0	0.2357	0.22825	0.3688	0.3421
p2	0.2357	0	0.20115	0.2042	0.1388
p36	0.22825	0.20115	0	0.18645	0.3382
p4	0.3688	0.2042	0.18645	0	0.2932
p5	0.3421	0.1388	0.3382	0.2932	0

Smallest distance from above is 0.1388

Hence, p2 and p5 forms second cluster.

	p1	p25	p36	p4
p1	0	0.2889	0.2347	0.3688
p25	0.2889	0	0.269675	0.2487
p36	0.2347	0.269675	0	0.18645
p4	0.3688	0.2487	0.18645	0

Smallest distance from above is 0.18645

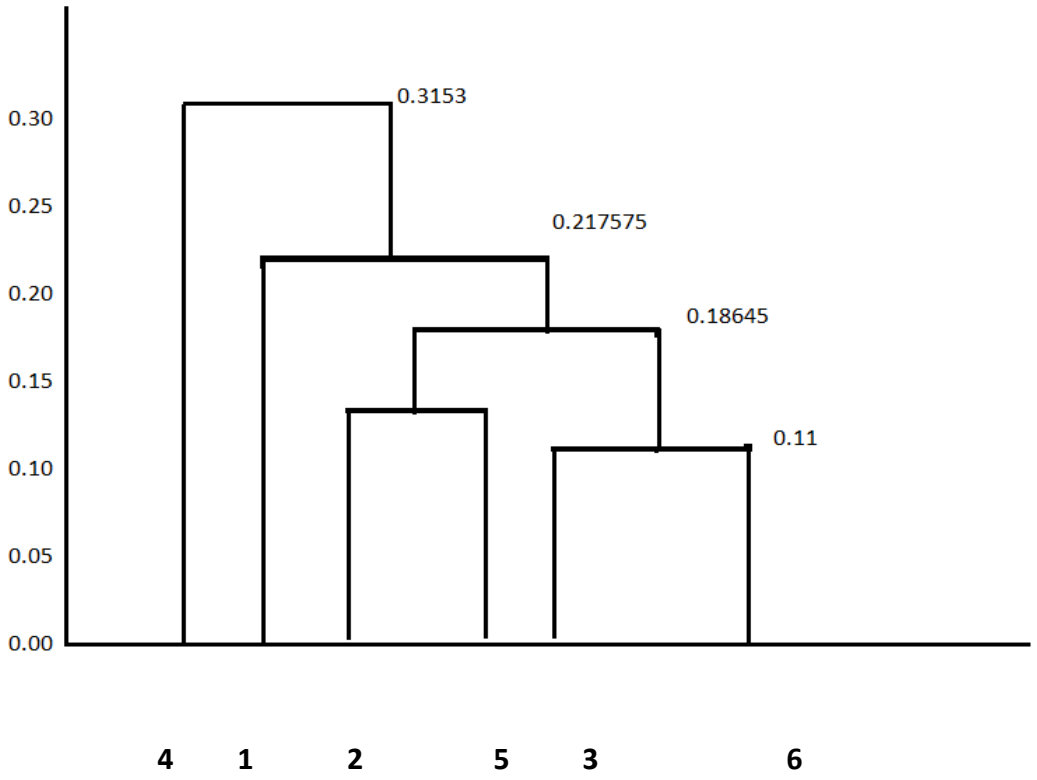
Hence, p36 and p25 forms third cluster.

	p1	p(25)(36)	p4
p1	0	0.2618	0.3688
p(25)(36)	0.2618	0	0.217575
p4	0.3688	0.217575	0

Smallest distance from above is 0.217575

Hence, p(25)(36) and p1 forms fourth cluster.

	p1(25)(36)	p4
p1(25)(36)	0	0.3153
p4	0.3153	0



Question2:

```
In [10]: dataframe.head()
```

Out[10]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUEN
0	C10001	40.900749	0.818182	95.40	0.00	95.4	0.000000	0.166
1	C10002	3202.467416	0.909091	0.00	0.00	0.0	6442.945483	0.000
2	C10003	2495.148862	1.000000	773.17	773.17	0.0	0.000000	1.000
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.0	205.788017	0.083
4	C10005	817.714335	1.000000	16.00	16.00	0.0	0.000000	0.083

CC GENERAL dataframe description.

```
In [11]: dataframe.describe()
```

Out[11]:

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY
count	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000
mean	1564.474828	0.877271	1003.204834	592.437371	411.067645	978.871112	0.490351
std	2081.531879	0.236904	2136.634782	1659.887917	904.338115	2097.163877	0.401371
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	128.281915	0.888889	39.635000	0.000000	0.000000	0.000000	0.083333
50%	873.385231	1.000000	361.280000	38.000000	89.000000	0.000000	0.500000
75%	2054.140036	1.000000	1110.130000	577.405000	468.637500	1113.821139	0.916667
max	19043.138560	1.000000	49039.570000	40761.250000	22500.000000	47137.211760	1.000000

Decription of dataframe

```
In [12]: df = dataframe.drop(['CUST_ID'], axis=1)
         df.head()
```

Out[12]:

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOI
0	40.900749	0.818182	95.40	0.00	95.4	0.000000	0.166667	
1	3202.467416	0.909091	0.00	0.00	0.0	6442.945483	0.000000	
2	2495.148862	1.000000	773.17	773.17	0.0	0.000000	1.000000	
3	1666.670542	0.636364	1499.00	1499.00	0.0	205.788017	0.083333	
4	817.714335	1.000000	16.00	16.00	0.0	0.000000	0.083333	

Drop the column cust_id

```
In [13]: df.isnull().any()
```

Out[13]:

```
BALANCE                             False
BALANCE_FREQUENCY                   False
PURCHASES                           False
ONEOFF_PURCHASES                    False
INSTALLMENTS_PURCHASES              False
CASH_ADVANCE                        False
PURCHASES_FREQUENCY                 False
ONEOFF_PURCHASES_FREQUENCY          False
PURCHASES_INSTALLMENTS_FREQUENCY    False
CASH_ADVANCE_FREQUENCY              False
CASH_ADVANCE_TRX                    False
PURCHASES_TRX                       False
CREDIT_LIMIT                         True
PAYMENTS                            False
MINIMUM_PAYMENTS                     True
PRC_FULL_PAYMENT                    False
TENURE                              False
dtype: bool
```

Check for any null values and replace them with the mean .

```
In [14]: df.fillna(dataframe.mean(), inplace=True)
         df.isnull().any()
```

Out[14]:

```
BALANCE                             False
BALANCE_FREQUENCY                   False
PURCHASES                           False
ONEOFF_PURCHASES                    False
INSTALLMENTS_PURCHASES              False
CASH_ADVANCE                        False
PURCHASES_FREQUENCY                 False
ONEOFF_PURCHASES_FREQUENCY          False
PURCHASES_INSTALLMENTS_FREQUENCY    False
CASH_ADVANCE_FREQUENCY              False
CASH_ADVANCE_TRX                    False
PURCHASES_TRX                       False
CREDIT_LIMIT                        False
PAYMENTS                            False
MINIMUM_PAYMENTS                    False
PRC_FULL_PAYMENT                    False
TENURE                              False
dtype: bool
```

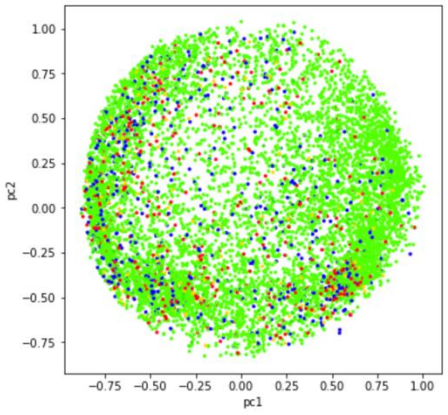
Performed scaling and pca with cluster k=2

Out[24]:

	P1	P2	TENURE
0	-0.488186	-0.677234	12
1	-0.517294	0.556074	12
2	0.334384	0.287313	12
3	-0.486617	-0.080780	12
4	-0.562175	-0.474770	12

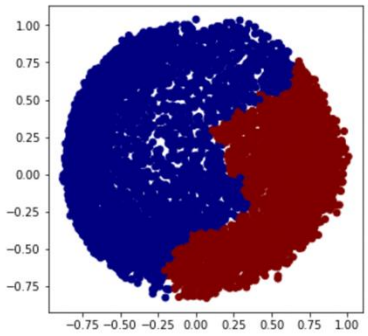
In [25]:
```
plt.figure(figsize=(6,6))
plt.scatter(finalDf['P1'],finalDf['P2'],c=finalDf['TENURE'],cmap='prism', s =5)
plt.xlabel('pc1')
plt.ylabel('pc2')
```

Out[25]:
```
Text(0, 0.5, 'pc2')
```



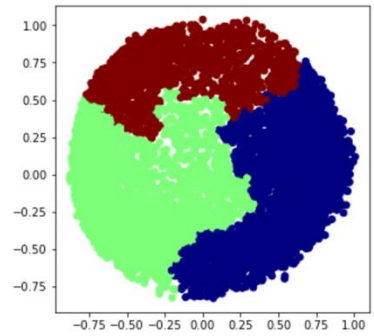
Visualizing the cluster k=2

In [28]:
```
ac2 = AgglomerativeClustering(n_clusters = 2)
# Visualizing the clustering
plt.figure(figsize =(5, 5))
plt.scatter(principalDf['P1'], principalDf['P2'],
           c = ac2.fit_predict(principalDf), cmap ='jet')
plt.show()
```



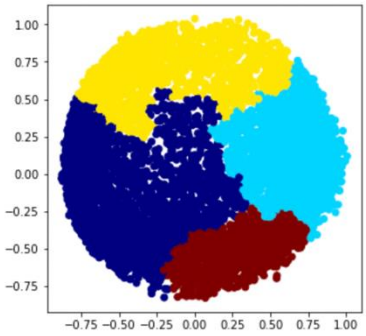
K=3

In [29]:
```
ac3 = AgglomerativeClustering(n_clusters = 3)
# Visualizing the cluster
plt.figure(figsize =(5, 5))
plt.scatter(principalDf['P1'], principalDf['P2'],
           c = ac3.fit_predict(principalDf), cmap ='jet')
plt.show()
```



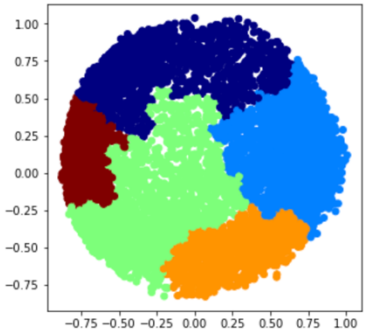
K=4

```
In [30]: ac4 = AgglomerativeClustering(n_clusters = 4)
         # Visualizing the cluster
         plt.figure(figsize =(5, 5))
         plt.scatter(principalDf['P1'], principalDf['P2'],
                     c = ac4.fit_predict(principalDf), cmap ='jet')
         plt.show()
```



K=5

```
In [31]: ac5 = AgglomerativeClustering(n_clusters = 5)
         # Visualizing the cluster
         plt.figure(figsize =(5, 5))
         plt.scatter(principalDf['P1'], principalDf['P2'],
                     c = ac5.fit_predict(principalDf), cmap ='jet')
         plt.show()
```



Appended the silhouette scores and plotted the bar graph as below: