Assignment – 6

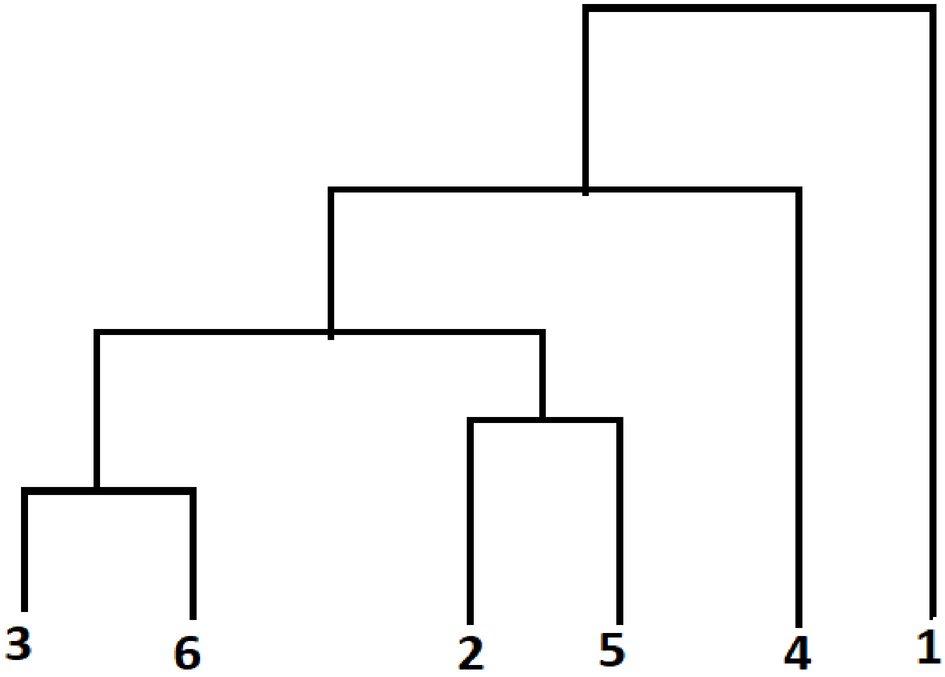
Name: Chethan Kacham
Username: CXK34890
Student ID: 700743489

Question - 1:
I have done all the calculations and plotted the dendrograms in question1.docx file. Please refer to the question.docx/question1.pdf file for question 1.

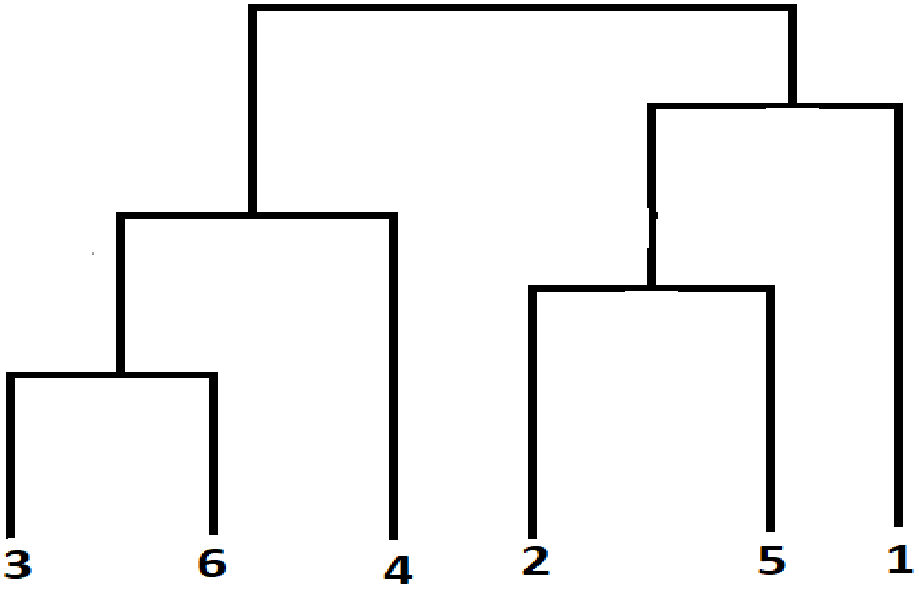
For first question, I have performed all the mathematical calculations for Single link, Complete link and Average link. Plotted the three dendrograms as shown below

For the Single link, I have used minimum distance to do the calculations and plot the dendrogram.
For the Complete link, I have used maximum distance to do the calculations and plot the dendrogram.
For the Average link, I have used average distance to do the calculations and plot the dendrogram.

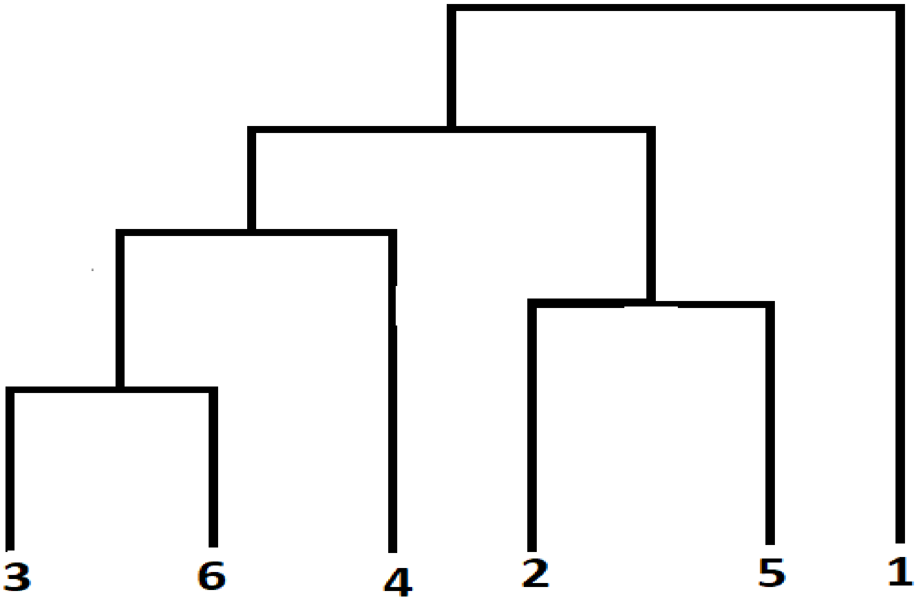
Dendrogram of Single link:



Dendrogram of Complete link:



Dendrogram of Average link:



Question - 2:

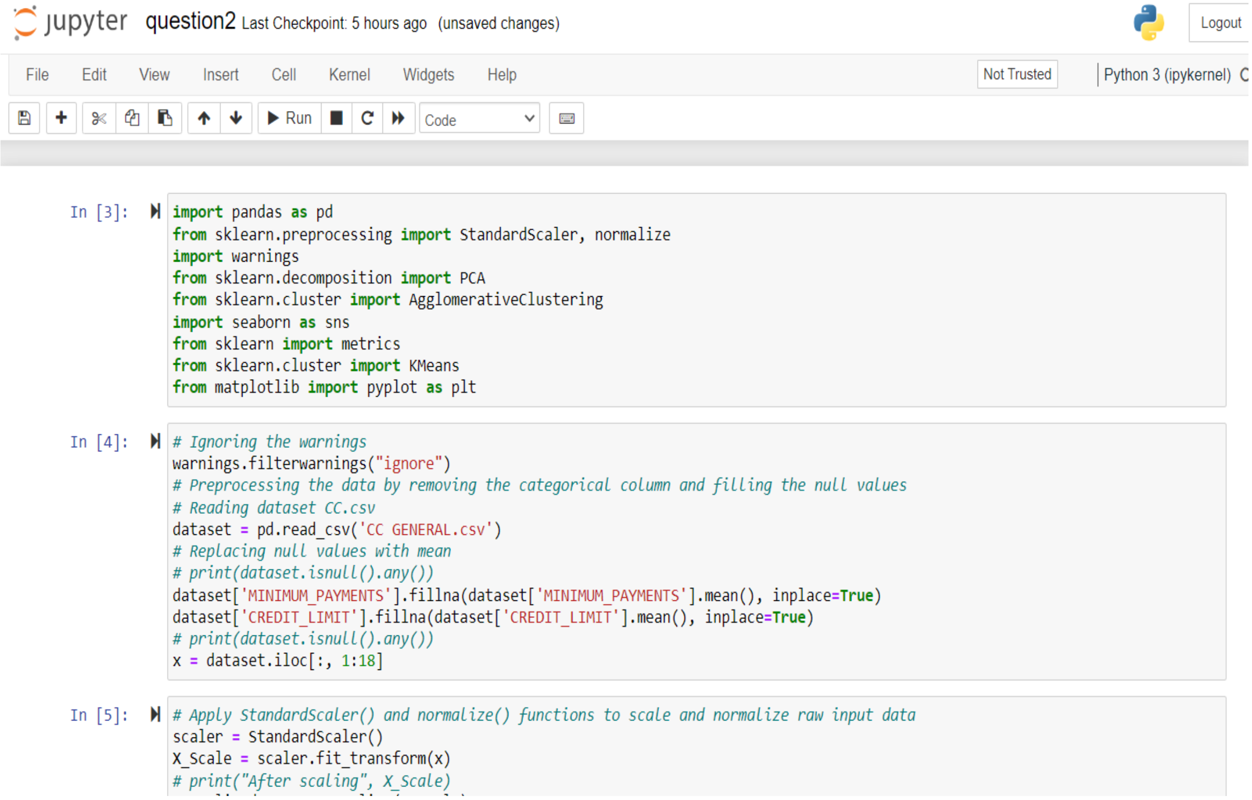
In the second question as explained in the class the followed the below steps

- Imported all the required modules and suppressed warnings.
- Read the data file and replaced the null values with mean and categorized the columns.
- Later applied StandardScaler and normalize functions.
- Used PCA(n=2) to reduce the input dimensions to two features.
- Applied Agglomerative Clustering with k=2,3,4 and 5 on finaldf.
- Visualized the results of Agglomerative clustering for k=2,3,4 and 5 using scatter plot.
- Evaluated the Silhouette scores for each k value i.e., 2,3,4 and 5 and plotted the results in a bar graph with respective to number of clusters.

I have commented the tasks as a heading in the code which are given to the question 2.

Adding the screenshots of the code and output for Question - 2

Code & Output:



In [3]:

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler, normalize
import warnings
from sklearn.decomposition import PCA
from sklearn.cluster import AgglomerativeClustering
import seaborn as sns
from sklearn import metrics
from sklearn.cluster import KMeans
from matplotlib import pyplot as plt
```

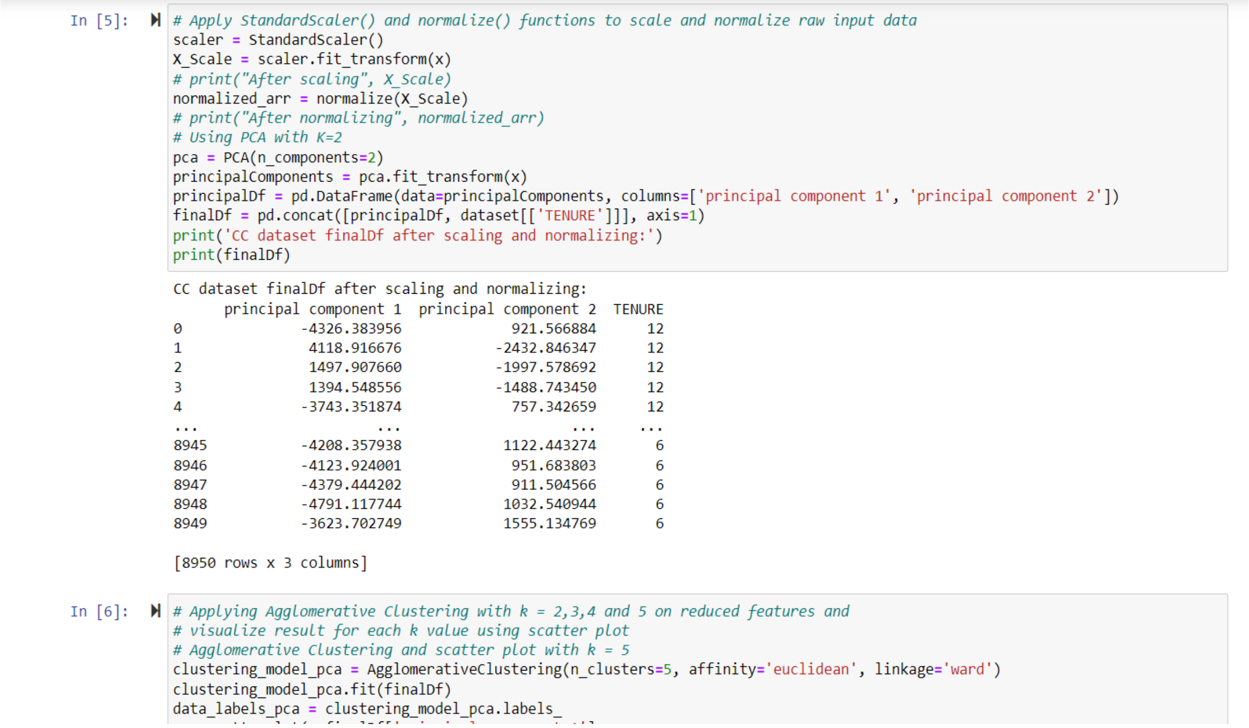
In [4]:

```python
# Ignoring the warnings
warnings.filterwarnings("ignore")
# Preprocessing the data by removing the categorical column and filling the null values
# Reading dataset CC.csv
dataset = pd.read_csv('CC GENERAL.csv')
# Replacing null values with mean
# print(dataset.isnull().any())
dataset['MINIMUM_PAYMENTS'].fillna(dataset['MINIMUM_PAYMENTS'].mean(), inplace=True)
dataset['CREDIT_LIMIT'].fillna(dataset['CREDIT_LIMIT'].mean(), inplace=True)
# print(dataset.isnull().any())
x = dataset.iloc[:, 1:18]
```

In [5]:

```python
# Apply StandardScaler() and normalize() functions to scale and normalize raw input data
scaler = StandardScaler()
X_Scale = scaler.fit_transform(x)
# print("After scaling", X_Scale)
```

Imported all the required modules and suppressed warnings, read the data file CC GENERAL.csv and replaced the null values with mean and categorized the columns in the above screenshot.

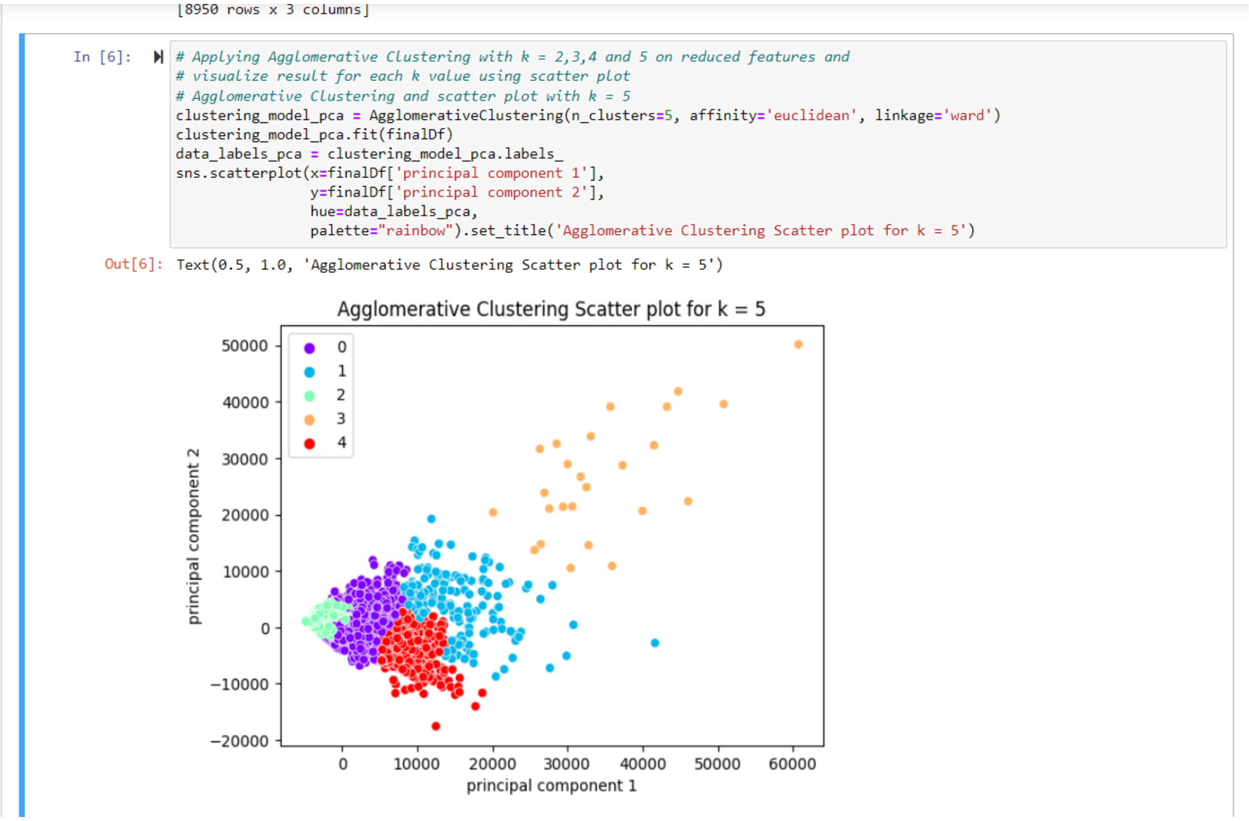


```
In [5]:  # Apply StandardScaler() and normalize() functions to scale and normalize raw input data
         scaler = StandardScaler()
         X_Scale = scaler.fit_transform(x)
         # print("After scaling", X_Scale)
         normalized_arr = normalize(X_Scale)
         # print("After normalizing", normalized_arr)
         # Using PCA with K=2
         pca = PCA(n_components=2)
         principalComponents = pca.fit_transform(x)
         principalDf = pd.DataFrame(data=principalComponents, columns=['principal component 1', 'principal component 2'])
         finalDf = pd.concat([principalDf, dataset[['TENURE']]], axis=1)
         print('CC dataset finalDf after scaling and normalizing:')
         print(finalDf)
```

```
CC dataset finalDf after scaling and normalizing:
      principal component 1  principal component 2  TENURE
0              -4326.383956             921.566884      12
1               4118.916676           -2432.846347      12
2               1497.907660           -1997.578692      12
3               1394.548556           -1488.743450      12
4              -3743.351874             757.342659      12
...                     ...                    ...     ...
8945           -4208.357938            1122.443274       6
8946           -4123.924001             951.683803       6
8947           -4379.444202             911.504566       6
8948           -4791.117744            1032.540944       6
8949           -3623.702749            1555.134769       6

[8950 rows x 3 columns]
```

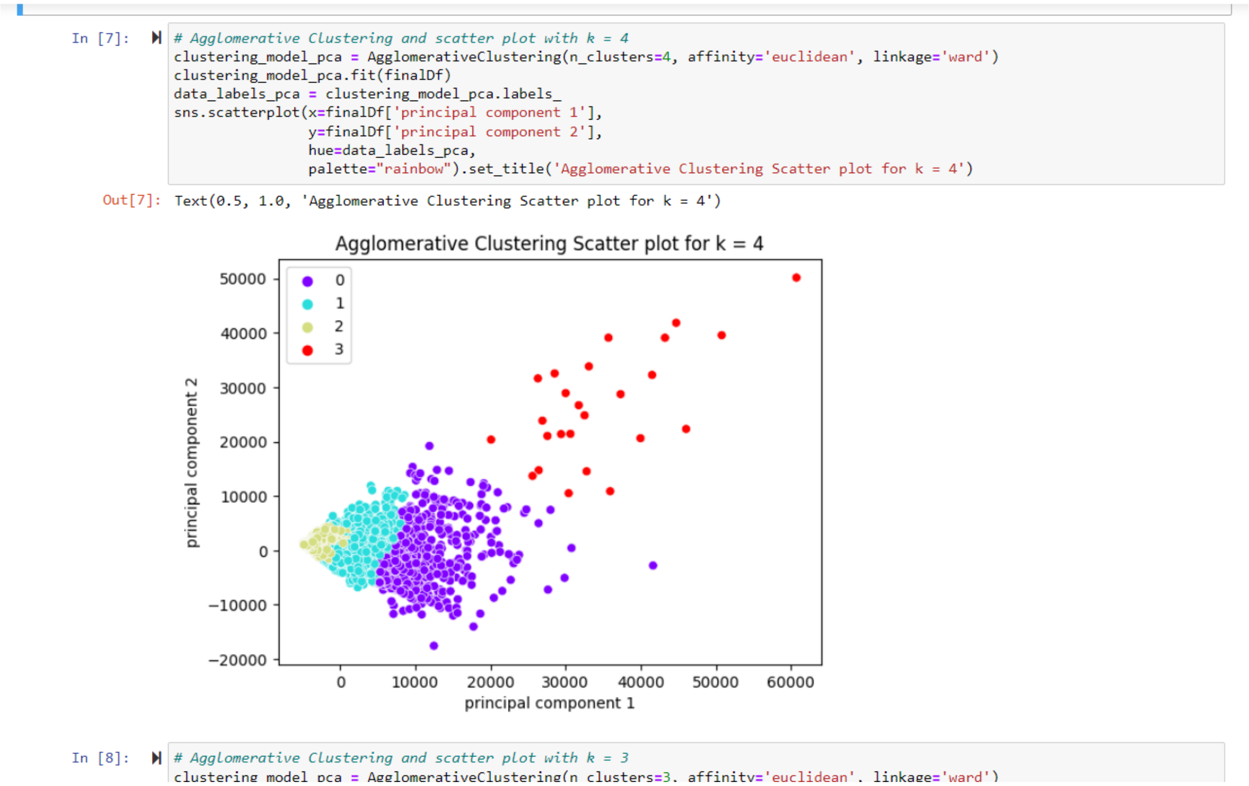
```
In [6]:  # Applying Agglomerative Clustering with k = 2,3,4 and 5 on reduced features and
         # visualize result for each k value using scatter plot
         # Agglomerative Clustering and scatter plot with k = 5
         clustering_model_pca = AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')
         clustering_model_pca.fit(finalDf)
         data_labels_pca = clustering_model_pca.labels_
```

Applied StandardScaler() function and normalize() function to the input data and then applied PCA(components=2) for reducing the data to two features in the above screenshot.



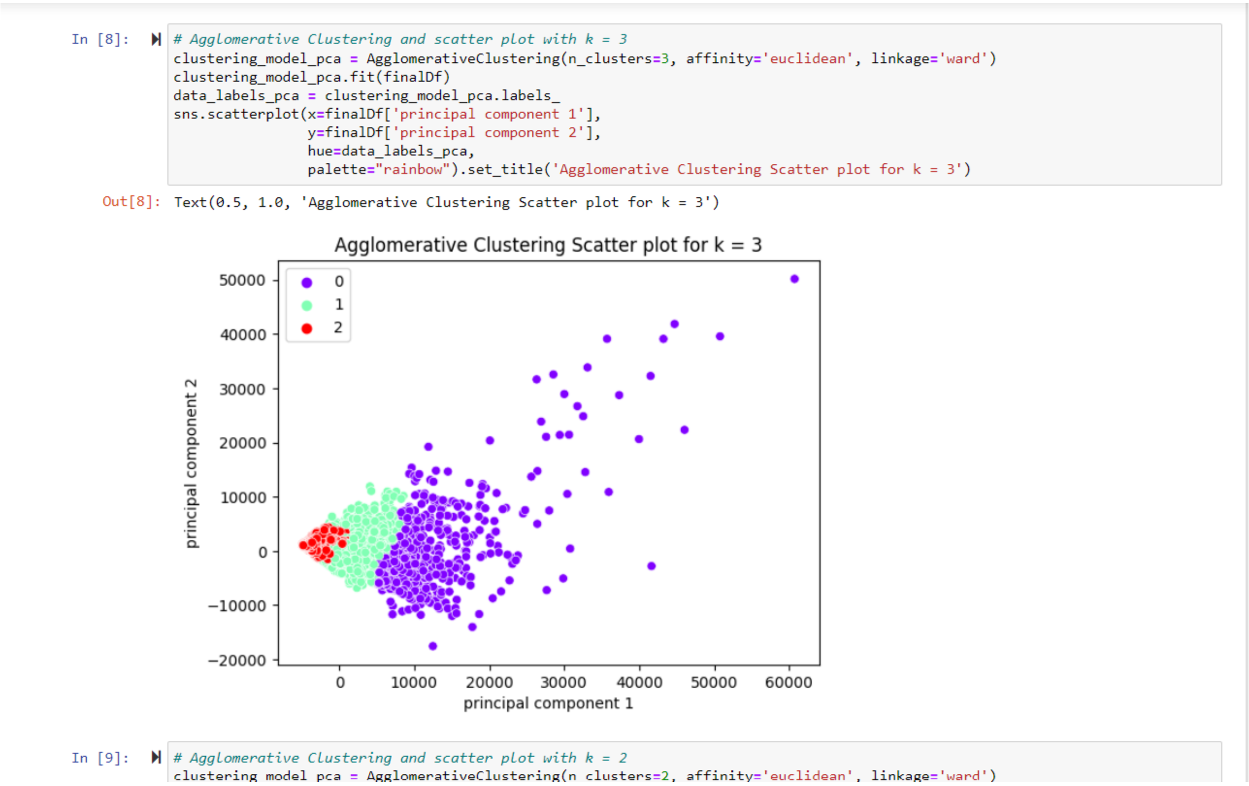
```
[8950 rows x 3 columns]
```

```
In [6]:  # Applying Agglomerative Clustering with k = 2,3,4 and 5 on reduced features and
         # visualize result for each k value using scatter plot
         # Agglomerative Clustering and scatter plot with k = 5
         clustering_model_pca = AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')
         clustering_model_pca.fit(finalDf)
         data_labels_pca = clustering_model_pca.labels_
         sns.scatterplot(x=finalDf['principal component 1'],
                         y=finalDf['principal component 2'],
                         hue=data_labels_pca,
                         palette="rainbow").set_title('Agglomerative Clustering Scatter plot for k = 5')

Out[6]: Text(0.5, 1.0, 'Agglomerative Clustering Scatter plot for k = 5')
```



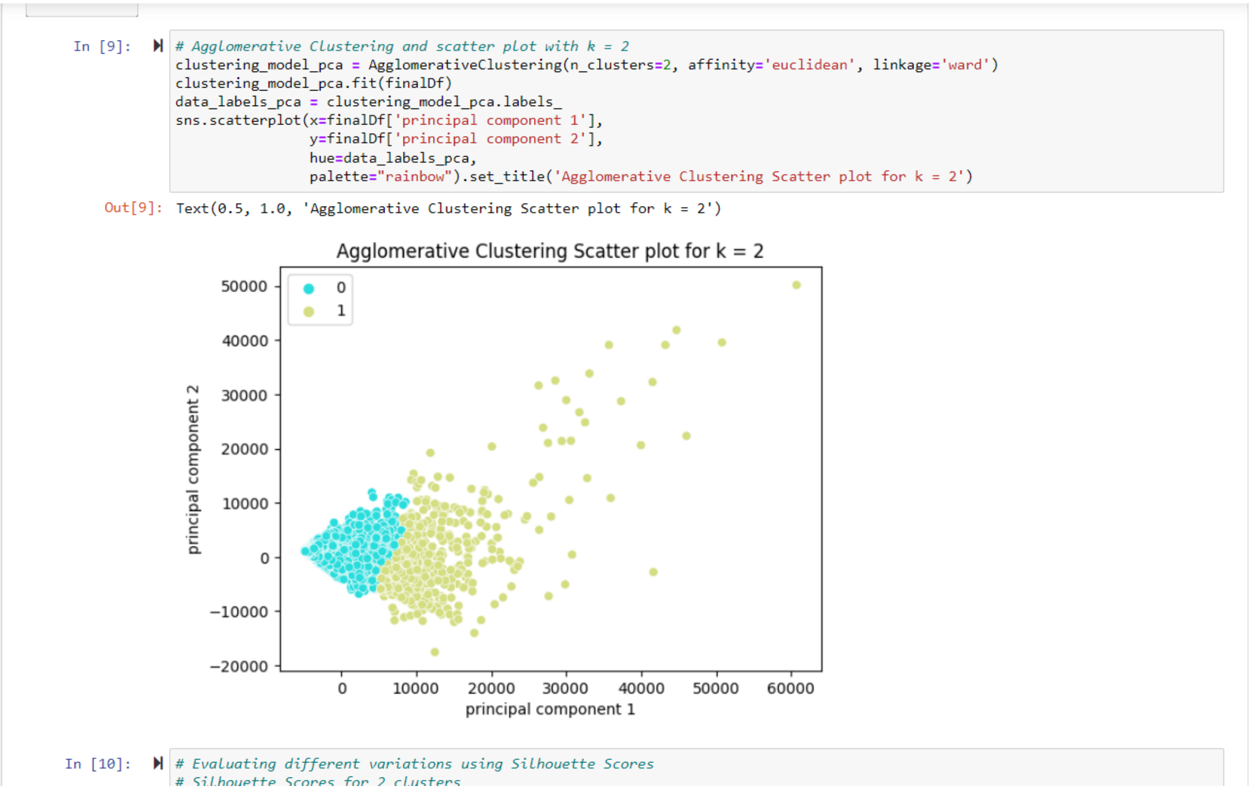
```
In [7]:  # Agglomerative Clustering and scatter plot with k = 4
         clustering_model_pca = AgglomerativeClustering(n_clusters=4, affinity='euclidean', linkage='ward')
         clustering_model_pca.fit(finalDf)
         data_labels_pca = clustering_model_pca.labels_
         sns.scatterplot(x=finalDf['principal component 1'],
                         y=finalDf['principal component 2'],
                         hue=data_labels_pca,
                         palette="rainbow").set_title('Agglomerative Clustering Scatter plot for k = 4')

Out[7]:  Text(0.5, 1.0, 'Agglomerative Clustering Scatter plot for k = 4')
```

```
In [8]:  # Agglomerative Clustering and scatter plot with k = 3
         clustering_model_pca = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
```



```
In [8]:  # Agglomerative Clustering and scatter plot with k = 3
         clustering_model_pca = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
         clustering_model_pca.fit(finalDf)
         data_labels_pca = clustering_model_pca.labels_
         sns.scatterplot(x=finalDf['principal component 1'],
                         y=finalDf['principal component 2'],
                         hue=data_labels_pca,
                         palette="rainbow").set_title('Agglomerative Clustering Scatter plot for k = 3')

Out[8]:  Text(0.5, 1.0, 'Agglomerative Clustering Scatter plot for k = 3')
```

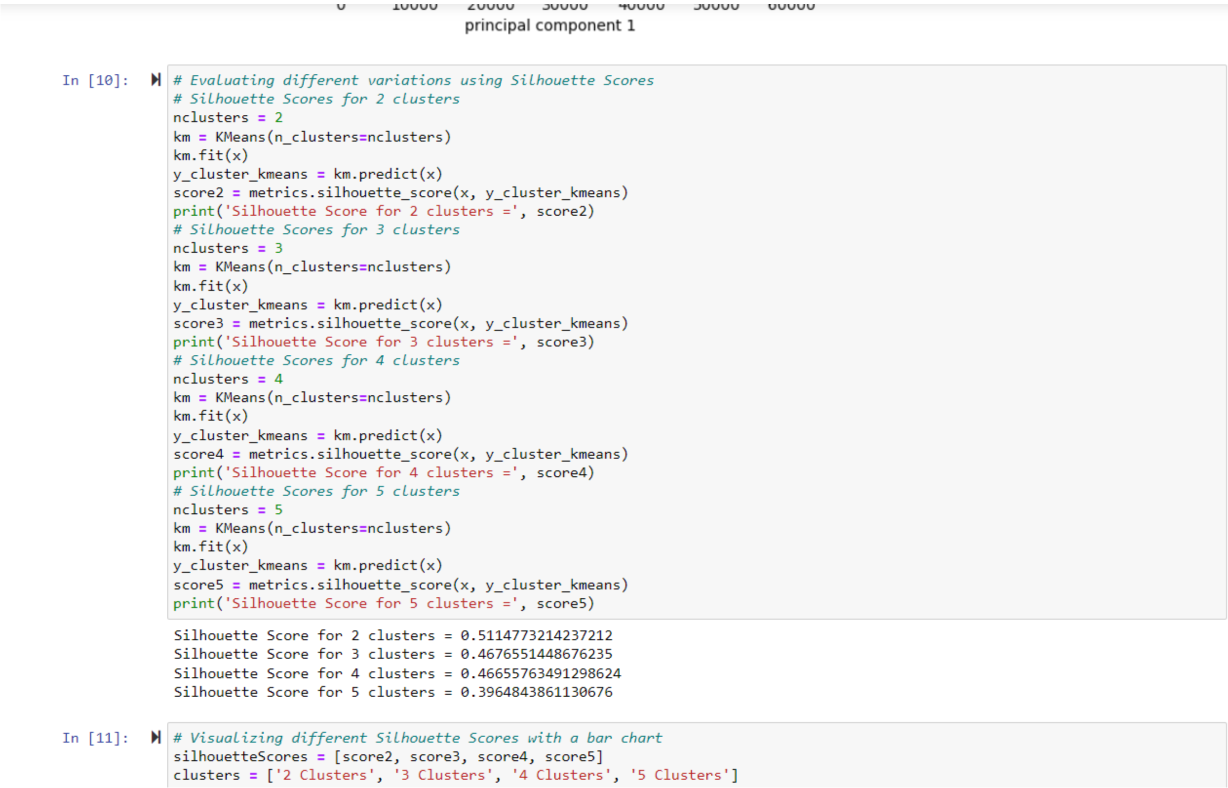
```
In [9]:  # Agglomerative Clustering and scatter plot with k = 2
         clustering_model_pca = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')
```



```
In [9]:  # Agglomerative Clustering and scatter plot with k = 2
         clustering_model_pca = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')
         clustering_model_pca.fit(finalDf)
         data_labels_pca = clustering_model_pca.labels_
         sns.scatterplot(x=finalDf['principal component 1'],
                         y=finalDf['principal component 2'],
                         hue=data_labels_pca,
                         palette="rainbow").set_title('Agglomerative Clustering Scatter plot for k = 2')

Out[9]: Text(0.5, 1.0, 'Agglomerative Clustering Scatter plot for k = 2')
```

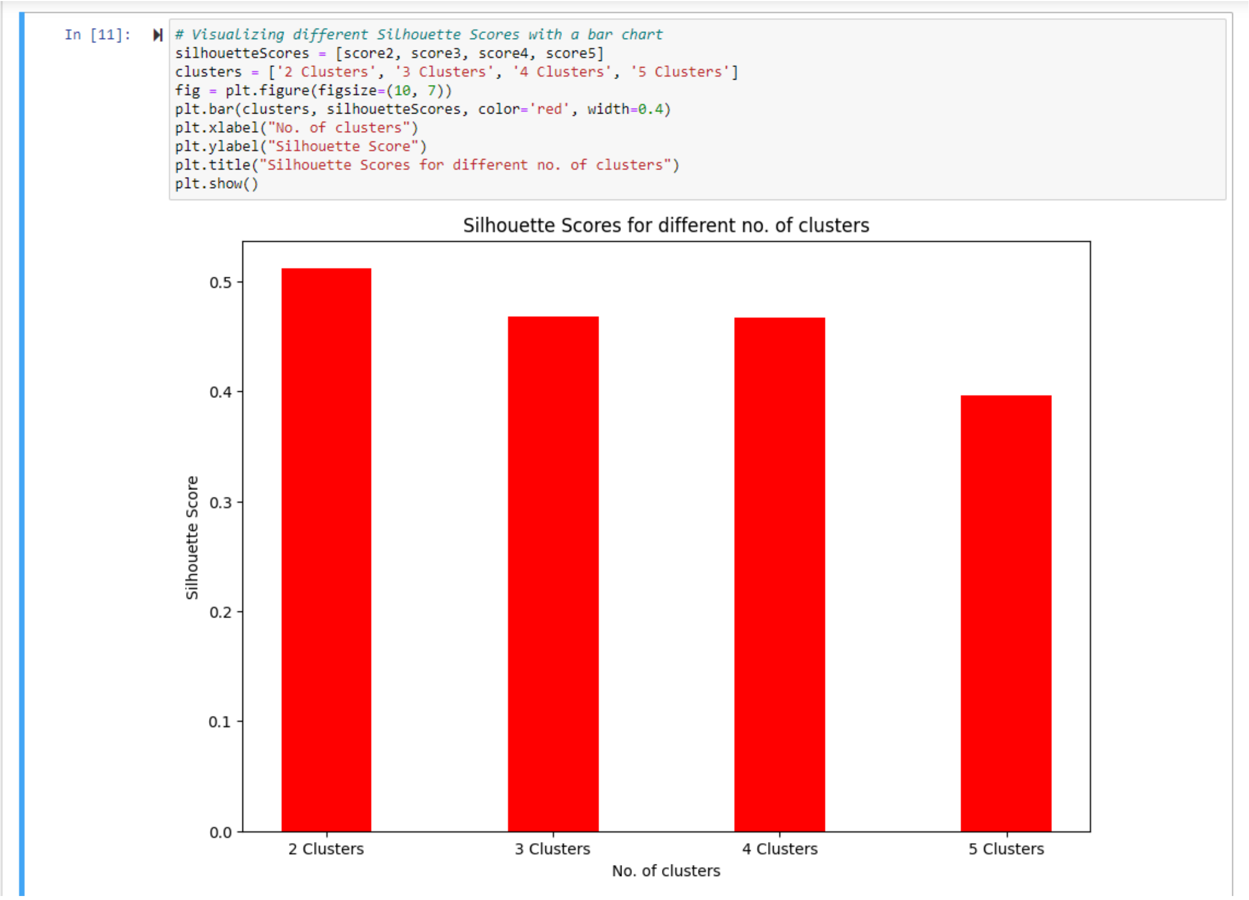
```
In [10]: # Evaluating different variations using Silhouette Scores
         # Silhouette Scores for 2 clusters
```

Applied Agglomerative Clustering with k=2,3,4 and 5 on finaldf and visualized the results using scatter plot in the above 4 screenshots.



```
In [10]: # Evaluating different variations using Silhouette Scores
         # Silhouette Scores for 2 clusters
         nclusters = 2
         km = KMeans(n_clusters=nclusters)
         km.fit(x)
         y_cluster_kmeans = km.predict(x)
         score2 = metrics.silhouette_score(x, y_cluster_kmeans)
         print('Silhouette Score for 2 clusters =', score2)
         # Silhouette Scores for 3 clusters
         nclusters = 3
         km = KMeans(n_clusters=nclusters)
         km.fit(x)
         y_cluster_kmeans = km.predict(x)
         score3 = metrics.silhouette_score(x, y_cluster_kmeans)
         print('Silhouette Score for 3 clusters =', score3)
         # Silhouette Scores for 4 clusters
         nclusters = 4
         km = KMeans(n_clusters=nclusters)
         km.fit(x)
         y_cluster_kmeans = km.predict(x)
         score4 = metrics.silhouette_score(x, y_cluster_kmeans)
         print('Silhouette Score for 4 clusters =', score4)
         # Silhouette Scores for 5 clusters
         nclusters = 5
         km = KMeans(n_clusters=nclusters)
         km.fit(x)
         y_cluster_kmeans = km.predict(x)
         score5 = metrics.silhouette_score(x, y_cluster_kmeans)
         print('Silhouette Score for 5 clusters =', score5)

         Silhouette Score for 2 clusters = 0.5114773214237212
         Silhouette Score for 3 clusters = 0.4676551448676235
         Silhouette Score for 4 clusters = 0.46655763491298624
         Silhouette Score for 5 clusters = 0.3964843861130676

In [11]: # Visualizing different Silhouette Scores with a bar chart
         silhouetteScores = [score2, score3, score4, score5]
         clusters = ['2 Clusters', '3 Clusters', '4 Clusters', '5 Clusters']
```

Evaluated the Silhouette Scores for k=2,3,4 and 5 in the above screenshot.



In [11]:

```
# Visualizing different Silhouette Scores with a bar chart
silhouetteScores = [score2, score3, score4, score5]
clusters = ['2 Clusters', '3 Clusters', '4 Clusters', '5 Clusters']
fig = plt.figure(figsize=(10, 7))
plt.bar(clusters, silhouetteScores, color='red', width=0.4)
plt.xlabel("No. of clusters")
plt.ylabel("Silhouette Score")
plt.title("Silhouette Scores for different no. of clusters")
plt.show()
```

Visualized the Silhouette Scores for different number of clusters using bar graph in the above screenshot.

In GitHub Machine Learning repository, we have an Assignment 6 folder where I have uploaded document(Word & PDF) for question 1 and I have uploaded the code for question 2 in question2.ipynb file. I have uploaded screenshots of dendrograms of single link, complete link and average link for question1 and screenshots of code and output for question 2 in Screenshots folder. As I have explained the code for question 2, I haven’t recorded any video for this assignment. I have uploaded word and pdf document of Assignment-6 in Word & PDF file of Assignment 6. The following is the link of the GitHub repository

GitHub link: https://github.com/ChethanKacham/MachineLearning.git

Video link: Explained the code in the class