Chitramoy_Mukherjee-DSC630-Week-04-Assignment 2

April 7, 2024

DSC-630-T302

Chitramoy Mukherjee

Date : 04/04/2024

Week3 - Exercise 4.2 - Clustering Exercise

```
[103]: from IPython.display import display, HTML
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA
```

Import the dataset in pandas dataframe and Display the first few rows of the dataset.

```
[59]: df = pd.read_csv('C:
  ↪\\Users\\Chitramoy\\Desktop\\MS-DSC\\DSC-630\\Week-4\\als_data.csv')
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2223 entries, 0 to 2222
Columns: 101 entries, ID to Urine.Ph_min
dtypes: float64(75), int64(26)
memory usage: 1.7 MB
None
```

```
[86]: # ID and SubjectID are the columns definitely not required for further analysis␣
  ↪process and can be consdered as irrelevant column.
irrelevant_columns = ['ID', 'SubjectID']

# Remove irrelevant columns
data = df.drop(columns=irrelevant_columns)

# Describe data set post cleanup.
```

```
data.describe()
```

```
[86]:          Age_mean  Albumin_max  Albumin_median  Albumin_min  Albumin_range  \
       count  2223.000000  2223.000000     2223.000000  2223.000000    2223.000000
       mean     54.550157    47.011134       43.952542    40.766347       0.013779
       std      11.396546     3.233980        2.654804     3.193087       0.009567
       min      18.000000    37.000000       34.500000    24.000000       0.000000
       25%      47.000000    45.000000       42.000000    39.000000       0.009042
       50%      55.000000    47.000000       44.000000    41.000000       0.012111
       75%      63.000000    49.000000       46.000000    43.000000       0.015873
       max      81.000000    70.300000       51.100000    49.000000       0.243902

              ALSFRS_slope  ALSFRS_Total_max  ALSFRS_Total_median  ALSFRS_Total_min  \
       count   2223.000000       2223.000000          2223.000000       2223.000000
       mean      -0.728274         31.692308            27.104926         19.877193
       std        0.622329          5.314228             6.633643          8.583509
       min       -4.345238         11.000000             2.500000          0.000000
       25%       -1.086310         29.000000            23.000000         14.000000
       50%       -0.620748         33.000000            28.000000         20.000000
       75%       -0.283832         36.000000            32.000000         27.000000
       max        1.207011         40.000000            40.000000         40.000000

              ALSFRS_Total_range  …  Sodium_median   Sodium_min  Sodium_range  \
       count         2223.000000  …    2223.000000  2223.000000   2223.000000
       mean             0.026035  …     140.145254   136.755061      0.015000
       std              0.016156  …       1.789886     2.715247      0.009283
       min              0.000000  …     128.000000   112.000000      0.000000
       25%              0.014035  …     139.000000   135.000000      0.010582
       50%              0.023297  …     140.000000   137.000000      0.013123
       75%              0.034799  …     141.000000   138.000000      0.017278
       max              0.117647  …     146.500000   145.000000      0.142857

                trunk_max  trunk_median    trunk_min  trunk_range  Urine.Ph_max  \
       count  2223.000000   2223.000000  2223.000000  2223.000000   2223.000000
       mean      6.203779      4.893387     2.955915     0.007136      6.820450
       std       1.747660      2.146076     2.358095     0.004503      0.932141
       min       0.000000      0.000000     0.000000     0.000000      5.000000
       25%       5.000000      3.000000     1.000000     0.003643      6.000000
       50%       7.000000      5.000000     3.000000     0.006920      7.000000
       75%       8.000000      6.500000     5.000000     0.009639      7.000000
       max       8.000000      8.000000     8.000000     0.042017      9.000000

              Urine.Ph_median  Urine.Ph_min
       count      2223.000000   2223.000000
       mean          5.710639      5.183221
       std           0.625039      0.437222
       min           5.000000      5.000000
```

```
25%           5.000000      5.000000
50%           6.000000      5.000000
75%           6.000000      5.000000
max           9.000000      8.000000

[8 rows x 99 columns]
```

[97]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Display basic information about the dataset
print(data.info())

# Apply StandardScaler to the numeric features
scaler = StandardScaler()
als_df_cleaned[numeric_features] = scaler.
 ↪fit_transform(als_df_cleaned[numeric_features])

als_df_cleaned.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2223 entries, 0 to 2222
Data columns (total 99 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Age_mean                        2223 non-null   int64
 1   Albumin_max                     2223 non-null   float64
 2   Albumin_median                  2223 non-null   float64
 3   Albumin_min                     2223 non-null   float64
 4   Albumin_range                   2223 non-null   float64
 5   ALSFRS_slope                    2223 non-null   float64
 6   ALSFRS_Total_max                2223 non-null   int64
 7   ALSFRS_Total_median             2223 non-null   float64
 8   ALSFRS_Total_min                2223 non-null   int64
 9   ALSFRS_Total_range              2223 non-null   float64
 10  ALT.SGPT._max                   2223 non-null   float64
 11  ALT.SGPT._median                2223 non-null   float64
 12  ALT.SGPT._min                   2223 non-null   float64
 13  ALT.SGPT._range                 2223 non-null   float64
 14  AST.SGOT._max                   2223 non-null   int64
 15  AST.SGOT._median                2223 non-null   float64
 16  AST.SGOT._min                   2223 non-null   float64
 17  AST.SGOT._range                 2223 non-null   float64
 18  Bicarbonate_max                 2223 non-null   float64
 19  Bicarbonate_median              2223 non-null   float64
 20  Bicarbonate_min                 2223 non-null   float64
 21  Bicarbonate_range               2223 non-null   float64
 22  Blood.Urea.Nitrogen..BUN._max   2223 non-null   float64
```

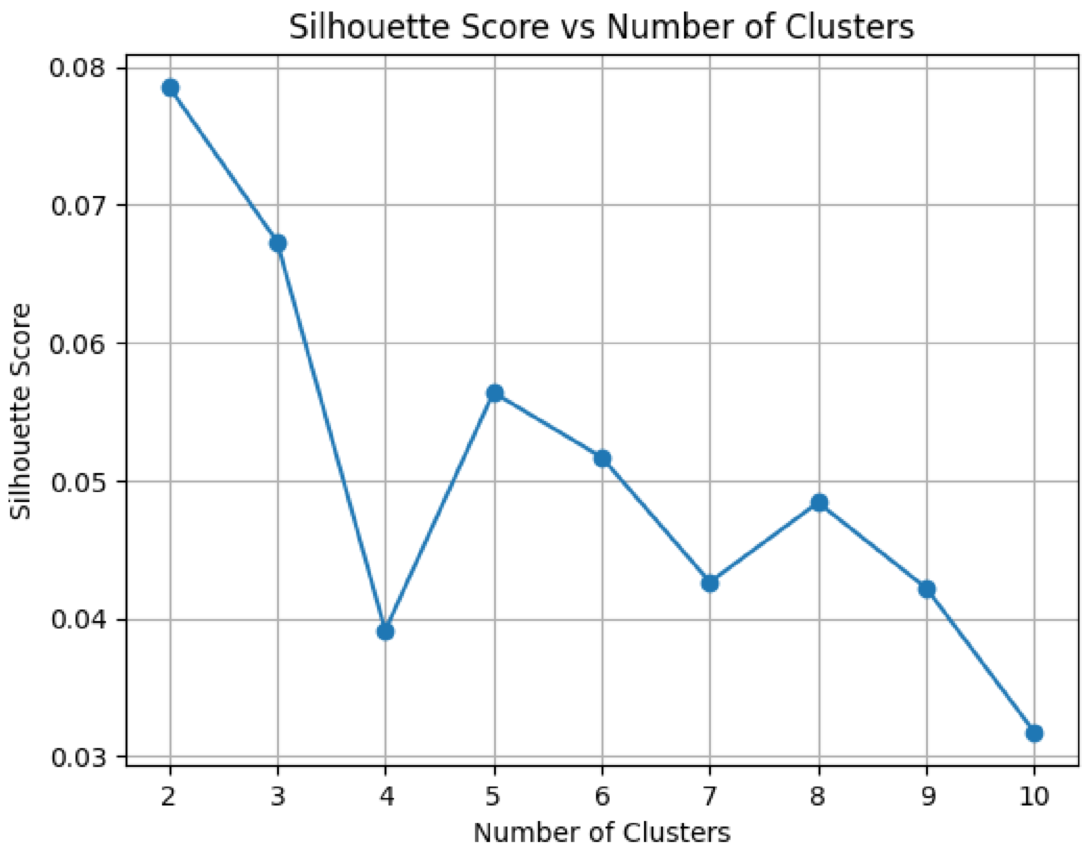
```
23  Blood.Urea.Nitrogen..BUN._median  2223 non-null   float64
24  Blood.Urea.Nitrogen..BUN._min     2223 non-null   float64
25  Blood.Urea.Nitrogen..BUN._range   2223 non-null   float64
26  bp_diastolic_max                  2223 non-null   int64
27  bp_diastolic_median               2223 non-null   float64
28  bp_diastolic_min                  2223 non-null   int64
29  bp_diastolic_range                2223 non-null   float64
30  bp_systolic_max                   2223 non-null   int64
31  bp_systolic_median                2223 non-null   float64
32  bp_systolic_min                   2223 non-null   int64
33  bp_systolic_range                 2223 non-null   float64
34  Calcium_max                       2223 non-null   float64
35  Calcium_median                    2223 non-null   float64
36  Calcium_min                       2223 non-null   float64
37  Calcium_range                     2223 non-null   float64
38  Chloride_max                      2223 non-null   float64
39  Chloride_median                   2223 non-null   float64
40  Chloride_min                      2223 non-null   float64
41  Chloride_range                    2223 non-null   float64
42  Creatinine_max                    2223 non-null   float64
43  Creatinine_median                 2223 non-null   float64
44  Creatinine_min                    2223 non-null   float64
45  Creatinine_range                  2223 non-null   float64
46  Gender_mean                       2223 non-null   int64
47  Glucose_max                       2223 non-null   float64
48  Glucose_median                    2223 non-null   float64
49  Glucose_min                       2223 non-null   float64
50  Glucose_range                     2223 non-null   float64
51  hands_max                         2223 non-null   int64
52  hands_median                      2223 non-null   float64
53  hands_min                         2223 non-null   int64
54  hands_range                       2223 non-null   float64
55  Hematocrit_max                    2223 non-null   float64
56  Hematocrit_median                 2223 non-null   float64
57  Hematocrit_min                    2223 non-null   float64
58  Hematocrit_range                  2223 non-null   float64
59  Hemoglobin_max                    2223 non-null   float64
60  Hemoglobin_median                 2223 non-null   float64
61  Hemoglobin_min                    2223 non-null   float64
62  Hemoglobin_range                  2223 non-null   float64
63  leg_max                           2223 non-null   int64
64  leg_median                        2223 non-null   float64
65  leg_min                           2223 non-null   int64
66  leg_range                         2223 non-null   float64
67  mouth_max                         2223 non-null   int64
68  mouth_median                      2223 non-null   float64
69  mouth_min                         2223 non-null   int64
70  mouth_range                       2223 non-null   float64
```

```
 71  onset_delta_mean                  2223 non-null   int64
 72  onset_site_mean                   2223 non-null   int64
 73  Platelets_max                     2223 non-null   int64
 74  Platelets_median                  2223 non-null   float64
 75  Platelets_min                     2223 non-null   float64
 76  Potassium_max                     2223 non-null   float64
 77  Potassium_median                  2223 non-null   float64
 78  Potassium_min                     2223 non-null   float64
 79  Potassium_range                   2223 non-null   float64
 80  pulse_max                         2223 non-null   int64
 81  pulse_median                      2223 non-null   float64
 82  pulse_min                         2223 non-null   int64
 83  pulse_range                       2223 non-null   float64
 84  respiratory_max                   2223 non-null   int64
 85  respiratory_median                2223 non-null   float64
 86  respiratory_min                   2223 non-null   int64
 87  respiratory_range                 2223 non-null   float64
 88  Sodium_max                        2223 non-null   float64
 89  Sodium_median                     2223 non-null   float64
 90  Sodium_min                        2223 non-null   float64
 91  Sodium_range                      2223 non-null   float64
 92  trunk_max                         2223 non-null   int64
 93  trunk_median                      2223 non-null   float64
 94  trunk_min                         2223 non-null   int64
 95  trunk_range                       2223 non-null   float64
 96  Urine.Ph_max                      2223 non-null   float64
 97  Urine.Ph_median                   2223 non-null   float64
 98  Urine.Ph_min                      2223 non-null   float64
dtypes: float64(75), int64(24)
memory usage: 1.7 MB
None
```

[97]:

```
   Age_mean  Albumin_max  Albumin_median  Albumin_min  Albumin_range  \
0  0.917137     3.089417       -1.300781    -0.866550       5.480929
1 -0.574879    -0.622016       -1.112401    -0.553303      -0.347725
2 -1.452535     0.924415        1.148162     1.326179      -0.507103
3  0.741606    -0.003443        0.017880     0.073191      -0.174361
4  0.741606    -0.003443        0.583021     0.386438      -0.573670

   ALSFRS_slope  ALSFRS_Total_max  ALSFRS_Total_median  ALSFRS_Total_min  \
0     -0.381450         -0.318520             0.134960          0.247368
1     -0.310907          0.998995             0.888863          0.130839
2     -0.299769         -1.447819            -1.975969         -1.150976
3      0.208801         -0.318520             0.285741          0.480425
4      0.456831          0.057913             0.059570          0.014311

   ALSFRS_Total_range  …  Sodium_median  Sodium_min  Sodium_range  \
0           -0.301588  …       2.992342    2.300470      0.260968
1            0.166537  …      -1.198812   -0.278144     -0.489913
2           -0.064100  …       1.595291    1.195350     -0.654169
3           -0.685524  …      -0.639992    0.458603     -0.272701
4           -0.350529  …      -0.081171    0.458603     -0.722774

   trunk_max  trunk_median  trunk_min  trunk_range  Urine.Ph_max  \
0   1.028018      0.981832   1.715365    -0.997420     -0.880376
1   1.028018      0.981832   0.867032    -0.388669      0.192665
2  -0.688950     -2.280669  -1.253800     0.398249     -0.880376
3  -0.688950      0.049689   0.018699    -0.477181      0.192665
4  -0.116627     -0.416383  -0.829634     0.300598     -0.880376

   Urine.Ph_median  Urine.Ph_min
0         0.463054      1.868532
1        -1.137208     -0.419151
2        -1.137208     -0.419151
3         0.463054     -0.419151
4        -1.137208     -0.419151

[5 rows x 99 columns]
```

[98]:
```python
# Initialize lists to store silhouette scores and cluster numbers
silhouette_scores = []
cluster_numbers = range(2, 11)  # Range of cluster numbers to try

# Iterate through each cluster number
for n_clusters in cluster_numbers:
    # Fit KMeans clustering model
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    cluster_labels = kmeans.fit_predict(als_df_cleaned)

    # Calculate silhouette score
    silhouette_avg = silhouette_score(als_df_cleaned, cluster_labels)
    silhouette_scores.append(silhouette_avg)

# Plot silhouette scores versus number of clusters
plt.plot(cluster_numbers, silhouette_scores, marker='o')
plt.xlabel('Number of Clusters')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Score vs Number of Clusters')
plt.xticks(cluster_numbers)
plt.grid(True)
plt.show()
```



[93]:
```
# Select the optimal number of clusters based on the plot
optimal_num_clusters = silhouette_scores.index(max(silhouette_scores)) + 2
print("Optimal number of clusters:", optimal_num_clusters)
```

```
Optimal number of clusters: 2
```

The cluster silhouette score is a metric used to evaluate the quality of clusters in K-means clustering. It provides a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette score ranges from -1 to 1, where:

A score close to +1 indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. A score around 0 indicates that the object is on or very close to the decision boundary between two neighboring clusters. A score close to -1 indicates that the object is likely to be assigned to the wrong cluster.

Based on the above plot 2 is the optimal number of cluster as that have the highest silhouette_scores value.

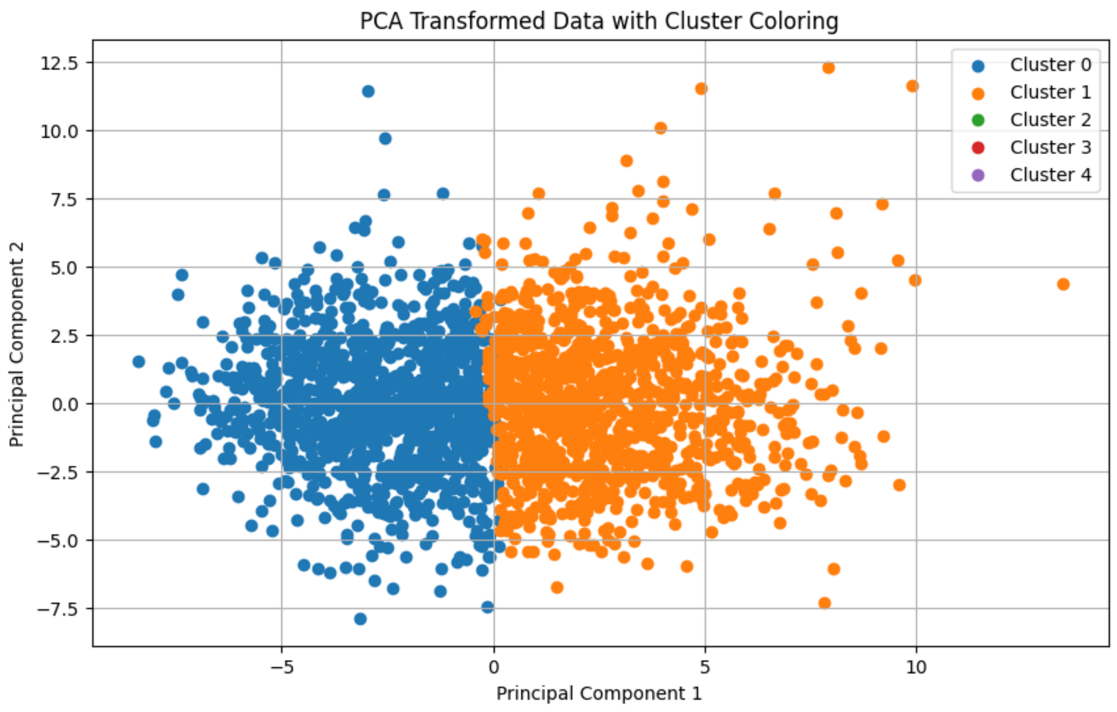
[100]:
```
# Optimal number of clusters chosen from silhouette score
optimal_num_clusters = 2

# Fit K-means model with optimal number of clusters
```

```
kmeans = KMeans(n_clusters=optimal_num_clusters, random_state=42)
cluster_labels = kmeans.fit_predict(als_df_cleaned)

# Add cluster labels to the original data
data['Cluster'] = cluster_labels
```

Loaded the dataset and preprocess it by scaling the data. Calculated the silhouette scores for different numbers of clusters and plot them against the number of clusters.Identified the optimal number of clusters based on the plot. Using K-means model calculate the optimal number of clusters.

[101]:
```
# Fit a PCA transformation with two features to the scaled data
pca = PCA(n_components=2)
pca_features = pca.fit_transform(scaled_data)
```

[102]:
```
# Create a DataFrame with PCA features and cluster labels
pca_df = pd.DataFrame(pca_features, columns=['PC1', 'PC2'])
pca_df['Cluster'] = cluster_labels

# Plot scatter plot with cluster coloring
plt.figure(figsize=(10, 6))
for cluster in range(5):  # Adjust based on the number of clusters
    cluster_data = pca_df[pca_df['Cluster'] == cluster]
    plt.scatter(cluster_data['PC1'], cluster_data['PC2'], label=f'Cluster␣
 ↪{cluster}')

plt.title('PCA Transformed Data with Cluster Coloring')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend()
plt.grid(True)
plt.show()
```



Above code performs all the specified tasks, including data preprocessing, determining the optimal number of clusters, fitting a K-means model, performing PCA transformation, and creating a scatterplot of the PCA-transformed data with cluster coloring.

The scatterplot visualizes the PCA-transformed data, where each point represents a sample from the dataset. The points are colored according to their assigned cluster labels obtained from the K-means clustering algorithm. By examining this plot, we can observe how well the clusters are separated in the reduced feature space (PCA components 1 and 2). If the clusters are well-separated, it indicates that the K-means algorithm has successfully grouped similar samples together. On the other hand, if the clusters overlap significantly, it suggests that the algorithm may not have effectively captured the underlying structure of the data. This visualization provides insights into the distribution and separation of clusters, aiding in the interpretation and evaluation of the clustering results.