4.2 Assignment. DSC630 - Jennifer Barrera Conde

June 28, 2024

1 DSC630

2 4.2 Exercise

3 Jennifer Barrera Conde

You will be using the dataset als_data.csv to apply clustering methods for this assignment. This data gives anonymized data on ALS patients. With this data, complete the following steps: Remove any data that is not relevant to the patient's ALS condition.

[ ]:

3.1 Load data set and Display for Preview

[1]:
```
import pandas as pd

# Load the dataset
file_path = 'als_data.csv'
als_data = pd.read_csv(file_path)

# Display the first few rows of the dataset to understand its structure
als_data.head()
```

[1]:
```
   ID  Age_mean  Albumin_max  Albumin_median  Albumin_min  Albumin_range  \
0   1        65         57.0            40.5         38.0       0.066202
1   2        48         45.0            41.0         39.0       0.010453
2   3        38         50.0            47.0         45.0       0.008929
3   4        63         47.0            44.0         41.0       0.012111
4   5        63         47.0            45.5         42.0       0.008292

   ALSFRS_slope  ALSFRS_Total_max  ALSFRS_Total_median  ALSFRS_Total_min  …  \
0     -0.965608                30                 28.0                22  …
1     -0.921717                37                 33.0                21  …
2     -0.914787                24                 14.0                10  …
3     -0.598361                30                 29.0                24  …
4     -0.444039                32                 27.5                20  …

   Sodium_min  Sodium_range  SubjectID  trunk_max  trunk_median  trunk_min  \
```

```
0       143.0      0.017422        533          8           7.0          7
1       136.0      0.010453        649          8           7.0          5
2       140.0      0.008929       1234          5           0.0          0
3       138.0      0.012469       2492          5           5.0          3
4       138.0      0.008292       2956          6           4.0          1

   trunk_range  Urine.Ph_max  Urine.Ph_median  Urine.Ph_min
0     0.002646           6.0              6.0           6.0
1     0.005386           7.0              5.0           5.0
2     0.008929           6.0              5.0           5.0
3     0.004988           7.0              6.0           5.0
4     0.008489           6.0              5.0           5.0

[5 rows x 101 columns]
```

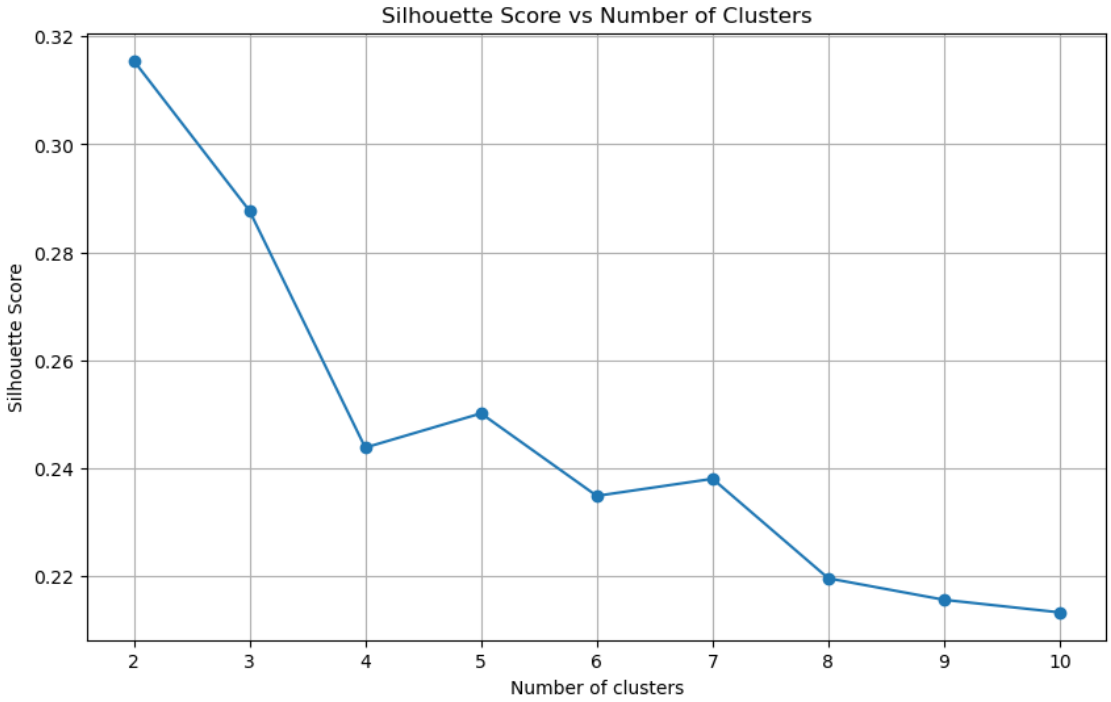
[ ]:

3.2 Step 1 - 3: Filter, Apply Standard Scaler, and Compute Silhouette Scores

[2]:
```python
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

# Select relevant columns
relevant_columns = ['Age_mean', 'ALSFRS_slope', 'ALSFRS_Total_max',␣
 ↪'ALSFRS_Total_median', 'ALSFRS_Total_min']
als_data_filtered = als_data[relevant_columns]

# Standardize the data
scaler = StandardScaler()
als_data_scaled = scaler.fit_transform(als_data_filtered)

# Calculate silhouette scores for different numbers of clusters
silhouette_scores = []
cluster_range = range(2, 11)

for n_clusters in cluster_range:
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    cluster_labels = kmeans.fit_predict(als_data_scaled)
    silhouette_avg = silhouette_score(als_data_scaled, cluster_labels)
    silhouette_scores.append(silhouette_avg)

# Plot the silhouette scores
plt.figure(figsize=(10, 6))
plt.plot(cluster_range, silhouette_scores, marker='o')
plt.title('Silhouette Score vs Number of Clusters')
```

```python
plt.xlabel('Number of clusters')
plt.ylabel('Silhouette Score')
plt.grid(True)
plt.show()
```

```
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



3.3 Step 4: Choose Optimal Number of Clusters

From the plot of silhouette scores versus the number of clusters, select the number of clusters that maximizes the silhouette score. This optimal number of clusters balances cluster cohesion and separation.

I have tried two methods to decided on my number of clusters. From the methods: *Peak Silhouette Score:* If the plot shows a peak silhouette score at 3 clusters, then 3 would be your optimal number of clusters.

Elbow Method: Sometimes the plot may not have a clear peak but has an elbow (a point where the silhouette score starts to decrease more slowly). The elbow point can also be a good choice for the optimal number of clusters.

I have found myself interested in either 2 or 3 clusters. I will show my results for both.

3.4 Step 5: Fit K-means Model with 2 Clusters

Using the optimal number of clusters as 2:

[3]:
```
# Fit the K-means model with the optimal number of clusters
optimal_clusters = 2
kmeans = KMeans(n_clusters=optimal_clusters, random_state=42)
kmeans.fit(als_data_scaled)
cluster_labels = kmeans.predict(als_data_scaled)
```

```
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

3.5 Step 6: Fit PCA Transformation

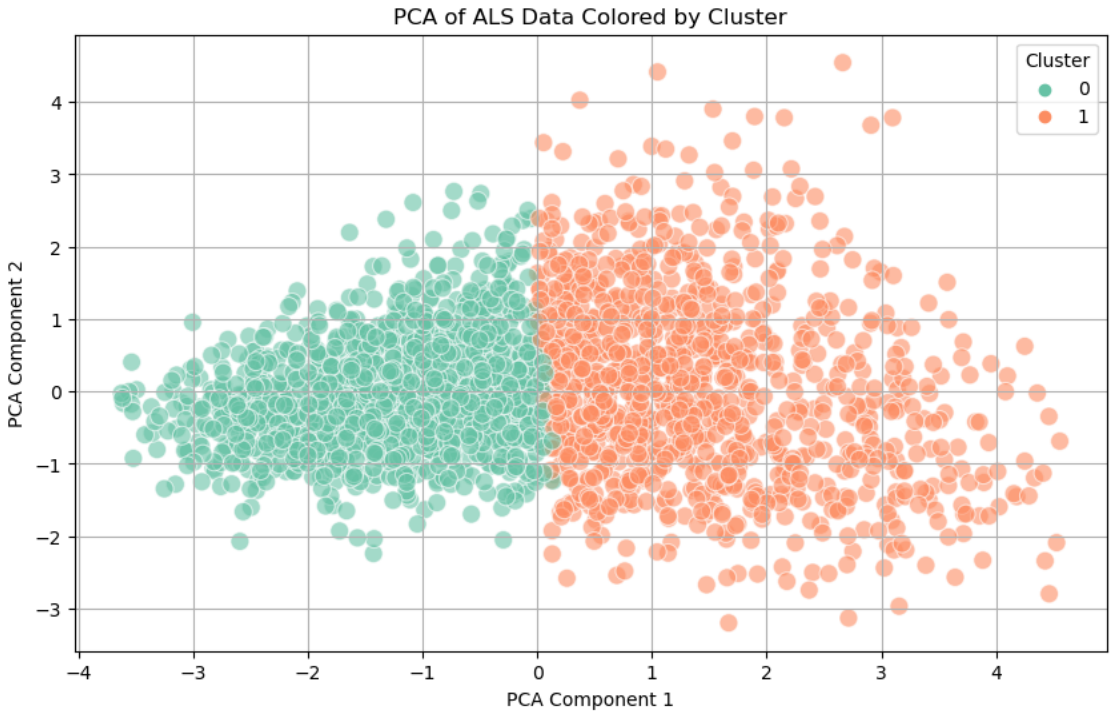
Perform PCA to reduce the dimensionality of the data for visualization:

[4]:
```
from sklearn.decomposition import PCA

# Fit PCA and transform the scaled data
pca = PCA(n_components=2)
pca_transformed = pca.fit_transform(als_data_scaled)

# Create a DataFrame with PCA results and cluster labels
pca_df = pd.DataFrame(data=pca_transformed, columns=['PCA1', 'PCA2'])
pca_df['Cluster'] = cluster_labels
```

3.6 Step 7: Scatter Plot of PCA-transformed Data

Visualize the PCA-transformed data, coloring each point by its cluster:

[5]:
```
import seaborn as sns

# Plot PCA-transformed data
plt.figure(figsize=(10, 6))
sns.scatterplot(data=pca_df, x='PCA1', y='PCA2', hue='Cluster', palette='Set2',␣
 ↪s=100, alpha=0.6)
plt.title('PCA of ALS Data Colored by Cluster')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.legend(title='Cluster')
plt.grid(True)
plt.show()
```



Cluster Separation: *Cluster 0 (Green):* This cluster occupies the left side of the plot, indicating that these data points have negative values for the first principal component. *Cluster 1 (Orange):* This cluster occupies the right side of the plot, indicating positive values for the first principal component.

There is a clear separation between the two clusters along the first principal component. Cluster 0 is largely on the left side, while Cluster 1 is on the right side, but there is some overlap around the center (near zero on the first principal component). This suggests that some patients might share characteristics of both clusters.

The two clusters represent distinct groups of ALS patients based on the selected features. These groups could correspond to different stages of disease progression or different patient profiles.

[ ]:

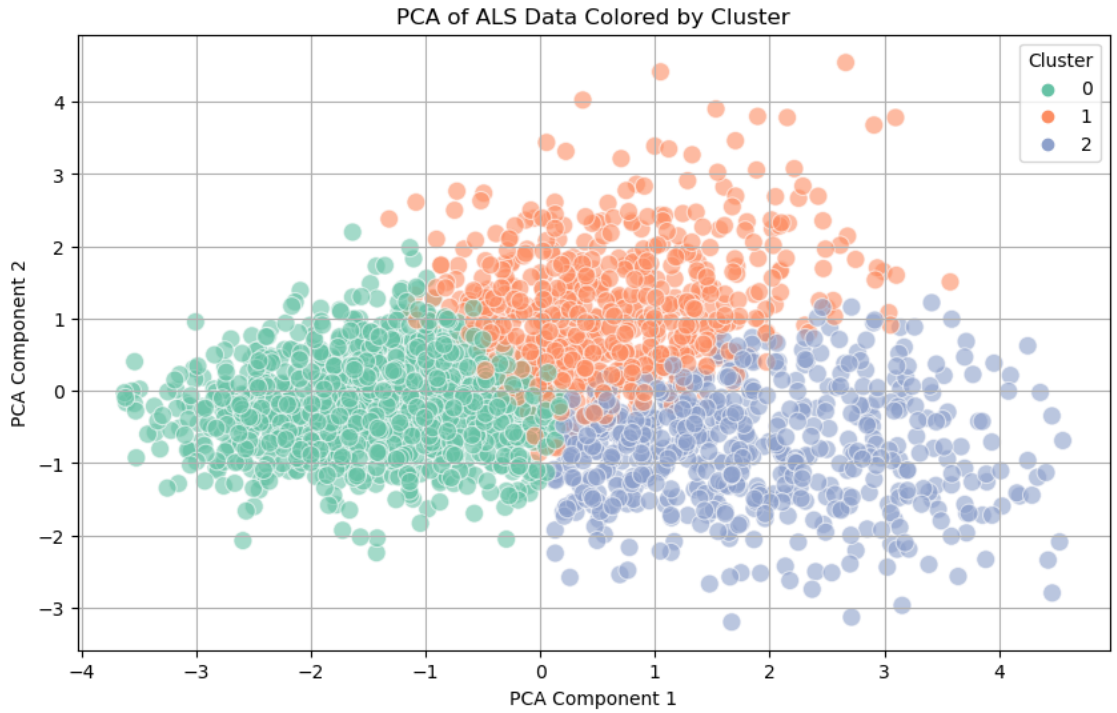
3.7 Step 5 through 7: Fit K-means Model with 3 Clusters

Using the optimal number of clusters as 3:

[6]:
```
# Fit the K-means model with the optimal number of clusters
optimal_clusters = 3
kmeans = KMeans(n_clusters=optimal_clusters, random_state=42)
kmeans.fit(als_data_scaled)
cluster_labels = kmeans.predict(als_data_scaled)

# Fit PCA and transform the scaled data
```

```
pca = PCA(n_components=2)
pca_transformed = pca.fit_transform(als_data_scaled)

# Create a DataFrame with PCA results and cluster labels
pca_df = pd.DataFrame(data=pca_transformed, columns=['PCA1', 'PCA2'])
pca_df['Cluster'] = cluster_labels

# Plot PCA-transformed data
plt.figure(figsize=(10, 6))
sns.scatterplot(data=pca_df, x='PCA1', y='PCA2', hue='Cluster', palette='Set2',␣
↪s=100, alpha=0.6)
plt.title('PCA of ALS Data Colored by Cluster')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.legend(title='Cluster')
plt.grid(True)
plt.show()
```

```
C:\Users\jbarr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



Cluster Separation: *Cluster 0 (Green):* This cluster occupies the left side of the plot, indicating that these data points have negative values for the first principal component. *Cluster 1 (Orange):*

This cluster is in the center of the plot, with data points spread around zero on the first principal component and positive values on the second principal component. *Cluster 2 (Blue):* This cluster occupies the right side of the plot, indicating positive values for the first principal component.

There is some overlap between all the clusters. This overlap suggests that there might be some similarities between the ALS patients in these clusters.

Cluster 1 appears to have a higher density of points in the center, suggesting that many patients fall into this intermediate group.

[ ]:

3.8 Summary and Conclusion:

Initially, the data appeared ambiguous and challenging to interpret due to my limited knowledge of ALS. To better understand the data's purpose and significance, I conducted thorough research on ALS and its clinical parameters.

Amyotrophic lateral sclerosis (ALS) is a fatal type of motor neuron disease. It causes progressive degeneration of nerve cells in the spinal cord and brain. It's often called Lou Gehrig disease after a famous baseball player who died from the disease. ALS is one of the most devastating types of disorders that affect nerve and muscle function.

ALS doesn't affect the senses (such as seeing or hearing). It also usually doesn't affect mental functioning. It isn't contagious. Currently, there is no cure for this disease.

ALS most often affects people between ages 40 and 70. But it can occur at a younger age. It affects people of all races and ethnic groups.

Symptoms are: Trouble walking or doing usual daily activities. Tripping and falling. Weakness in the legs, feet or ankles. Hand weakness or clumsiness. Slurred speech or trouble swallowing. Weakness associated with muscle cramps and twitching in the arms, shoulders and tongue. Untimely crying, laughing or yawning. Thinking or behavioral changes.

Reference: https://www.hopkinsmedicine.org/health/conditions-and-diseases/amyotrophic-lateral-sclerosis-als

https://www.mayoclinic.org/diseases-conditions/amyotrophic-lateral-sclerosis/symptoms-causes/syc-20354022

With this newfound understanding, along with additional information I gathered, I was able to proceed with the assignment with a clearer comprehension of the data and its implications.

3.8.1 For 2 Clusters:

Cluster Characteristics: *Cluster 0:* Patients in this cluster might have lower values in the selected features, such as lower ALSFRS scores, suggesting a more advanced stage of ALS. *Cluster 1:* Patients in this cluster could have higher values for the selected features, possibly indicating an earlier stage of ALS or better overall health.

The PCA plot provides a visual representation of the ALS patient data clustered into two distinct groups. The separation of clusters along the first principal component suggests significant differences between the two groups, which could correspond to different ALS stages or subtypes. Further

clinical interpretation and validation are needed to confirm these findings and their implications for ALS treatment and management. The overlap near the center indicates that some patients share characteristics of both clusters, highlighting the complexity and variability of ALS.

3.8.2 For 3 Clusters:

Cluster Characteristics: *Cluster 0:* Patients in this cluster might have lower values in the selected features, such as lower ALSFRS scores, suggesting a more advanced stage of ALS. *Cluster 1:* Patients in this cluster could be in an intermediate stage, with moderate values for the features. *Cluster 2:* Patients in this cluster might have higher values for the selected features, possibly indicating an earlier stage of ALS or better overall health.

Further Analysis: Further analysis could involve examining the mean and variance of the original features within each cluster to understand the characteristics of each group better. Clinical interpretation and validation are crucial to determine if these clusters align with known stages or subtypes of ALS.

The PCA plot provides a visual representation of the ALS patient data clustered into three groups. The separation and overlap of clusters can help identify distinct patient profiles or stages of disease progression. Further clinical interpretation and validation are needed to confirm these findings and their implications for ALS treatment and management.

[ ]: