In [1]:

```
# William Barker
# DSC630
# Week 4

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv('als_data.csv')
df.head()
```

Out[2]:

	ID	Age_mean	Albumin_max	Albumin_median	Albumin_min	Albumin_range	ALSFRS_slope	ALSFRS_Tot
0	1	65	57.0	40.5	38.0	0.066202	-0.965608	
1	2	48	45.0	41.0	39.0	0.010453	-0.921717	
2	3	38	50.0	47.0	45.0	0.008929	-0.914787	
3	4	63	47.0	44.0	41.0	0.012111	-0.598361	
4	5	63	47.0	45.5	42.0	0.008292	-0.444039	

5 rows × 101 columns

In [3]:

```
# Remove irrelevant columns
relevant_columns = ['Age_mean', 'Albumin_range', 'ALSFRS_slope', 'ALSFRS_Total_range', 'Cr
df = df[relevant_columns]
df
```

Out[3]:

	Age_mean	Albumin_range	ALSFRS_slope	ALSFRS_Total_range	Creatinine_range
0	65	0.066202	-0.965608	0.021164	0.030801
1	48	0.010453	-0.921717	0.028725	0.030801
2	38	0.008929	-0.914787	0.025000	0.031571
3	63	0.012111	-0.598361	0.014963	0.044090
4	63	0.008292	-0.444039	0.020374	0.058640
...	...	...	...	...	...
2218	33	0.008772	-0.239501	0.009107	0.046526
2219	61	0.009074	-0.388711	0.025408	0.056261
2220	47	0.012111	-0.108631	0.010949	0.048654
2221	37	0.017857	-0.855880	0.023214	0.063143
2222	48	0.018476	-2.050562	0.059908	0.059363

2223 rows × 5 columns

In [4]:

```
from sklearn.preprocessing import StandardScaler

# Initialize the scaler
scaler = StandardScaler()
```

```
# Scale the data
scaled_data = scaler.fit_transform(df)
```

In [5]:

```
pip install threadpoolctl==3.1.0
```

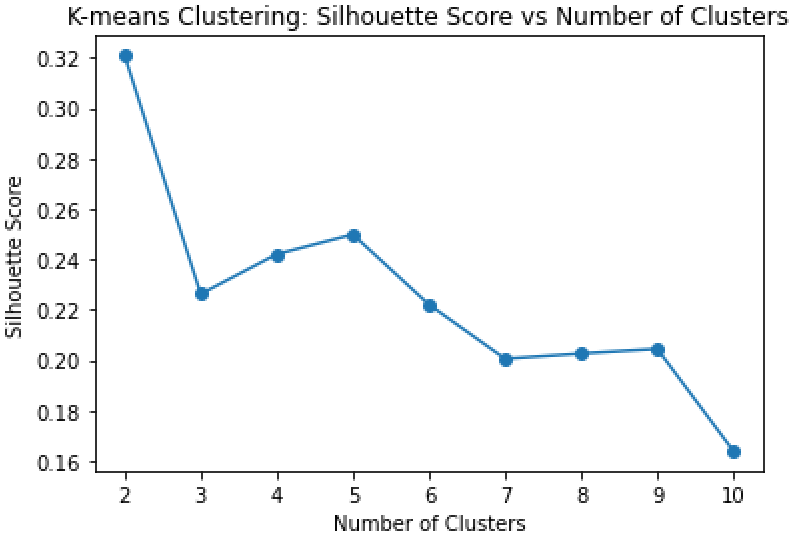
```
Requirement already satisfied: threadpoolctl==3.1.0 in ./opt/anaconda3/lib/python3.9/site-
packages (3.1.0)
Note: you may need to restart the kernel to use updated packages.
```

In [6]:

```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

# Initialize lists to store silhouette scores and number of clusters
silhouette_scores = []
num_clusters = []

# Try different numbers of clusters
for k in range(2, 11):
    # Fit K-means clustering model
    kmeans = KMeans(n_clusters=k, random_state=42)
    labels = kmeans.fit_predict(scaled_data)

    # Calculate silhouette score
    score = silhouette_score(scaled_data, labels)

    # Append scores and number of clusters
    silhouette_scores.append(score)
    num_clusters.append(k)

# Create plot
plt.plot(num_clusters, silhouette_scores, marker='o')
plt.xlabel('Number of Clusters')
plt.ylabel('Silhouette Score')
plt.title('K-means Clustering: Silhouette Score vs Number of Clusters')
plt.show()
```

```
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
```

```
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```



In [10]:

```
# Set the optimal number of clusters
# two had the highest silhouette score so we are gonna go with two
optimal_num_clusters = 2

# Fit K-means clustering model with optimal number of clusters
kmeans = KMeans(n_clusters=optimal_num_clusters, random_state=42)
labels = kmeans.fit_predict(scaled_data)
```

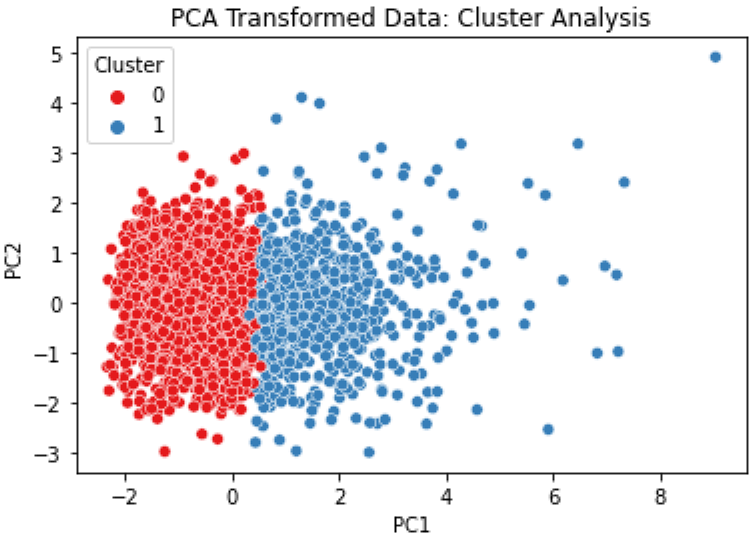
```
/Users/cameronbarker/opt/anaconda3/lib/python3.9/site-packages/sklearn/cluster/_kmeans.py:
870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Se
t the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

In [11]:

```
from sklearn.decomposition import PCA

# Initialize PCA model with 2 components
pca = PCA(n_components=2)

# Perform PCA transformation on the scaled data
pca_transformed = pca.fit_transform(scaled_data)
```

In [12]:

```
import seaborn as sns

# Create DataFrame for plotting
pca_df = pd.DataFrame({'PC1': pca_transformed[:, 0], 'PC2': pca_transformed[:, 1], 'Cluste

# Create scatterplot with cluster coloring
sns.scatterplot(data=pca_df, x='PC1', y='PC2', hue='Cluster', palette='Set1')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('PCA Transformed Data: Cluster Analysis')
plt.show()
```



In [ ]:

```
# Summary
# Our silhouette score visualization made it easy for us to decide how many clusters to i
# Because our number of clusters was so small, we can easily identify outliers, which in
# represented in our blue cluster.
```