Assignment 2: Clustering

[Github]

PROJECT DOCUMENTATION

DATE: 28/04/2024

PROJECT TITLE: Clustering Patient Charges: K-Medoid & DB Scan Showcase

OBJECTIVE: Aiming to analyze a dataset containing information about patients, including their age, sex, BMI, number of children, smoker status, region, and medical charges. The dataset will be used to cluster patients, using Python, based on similarities in their demographic and lifestyle factors, specifically focusing on sex, smoker status, and charges.

GIVEN TASKS:

1. **Clustering Tendency Test (Hopkins Statistic):**

```
hopkin=hopkins(df)
print('Hopkins measure: ',hopkin)
```

```
Hopkins measure:  0.9357020002439028
```

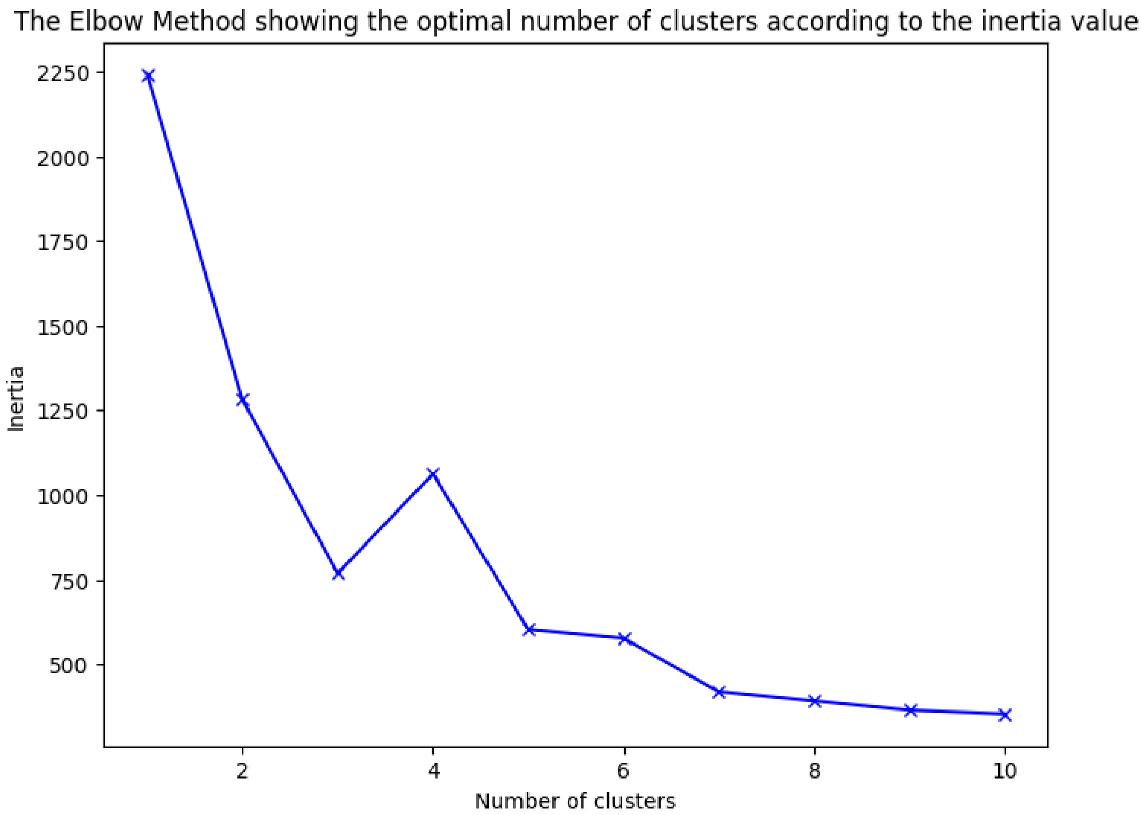
A Hopkins measure of 0.9357 indicates a strong clustering tendency in the dataset. This suggests that the data points are not randomly distributed and are likely to form meaningful clusters when using clustering algorithms like k-medoids or DBSCAN.

---

2. **K Clusters needed in K-Medoid using the Elbow Method:**

```
for n in num_clusters:
    kmedoids = KMedoids(n_clusters=n, random_state=0)
    kmedoids.fit(X_scaled)
    inertia_list.append(kmedoids.inertia_)
plt.figure(figsize=(8, 6))
plt.plot(num_clusters, inertia_list, 'bx-')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.title('The Elbow Method showing the optimal number of clusters according to the inertia value')
plt.show()
```

From the graph it's observed that the optimal number of clusters according to inertia is: 3-4 clusters.



---

3. K-Medoid Clustering + Visualizations + Clustering Quality:

- Library needed to import k-Medoid:

```
pip install scikit-learn-extra
```

- Measuring the Inertia:

```
for n_clusters in range(2, 10):
    for random_state in range(10):
        kmedoids = KMedoids(n_clusters=n_clusters,
random_state=random_state)
        kmedoids.fit(X_scaled)
        if kmedoids.inertia_ < best_inertia:
            best_inertia = kmedoids.inertia_
            best_kmedoids = kmedoids
```

```
print("Best inertia: ", best_inertia)
```

The lower the inertai the higher the cohesiviness, the best inertia resulted was:

```
Best inertia:  365.13885745548237
```

- Measuring metrics for this technique:

```
silhouette = silhouette_score(X_scaled, cluster_labels)
calinski_harabasz = calinski_harabasz_score(X_scaled, cluster_labels)
davies_bouldin = davies_bouldin_score(X_scaled, cluster_labels)
```

Resulting:

```
Silhouette Coefficient: 0.5329259467802452
Calinski-Harabasz Score: 2084.556362028302
Davies-Bouldin Score: 0.6241262044242547
```

Which implies that our data will have moderate clustering reults with a resonable seperation.

- Visualizing K-Medoid
  - Selected Features:

```
# Select the features to use for clustering
selected_features = ['sex', 'smoker', 'charges']
```

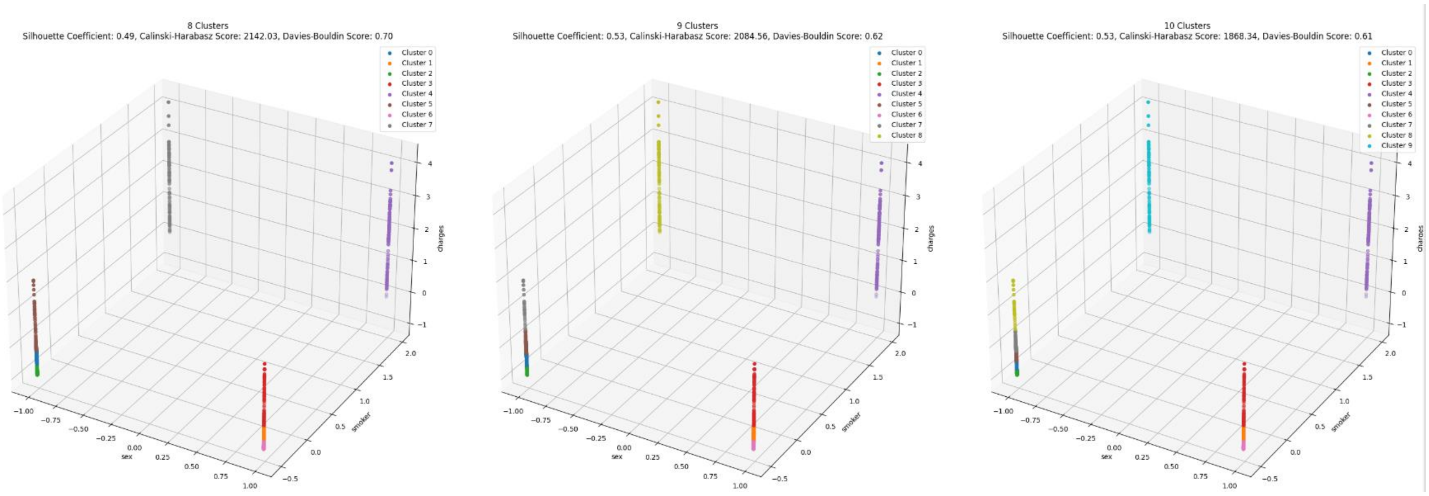
  - 3D Visulaizer:

```
fig, axes = plt.subplots(3, 3, figsize=(18, 18),
subplot_kw={'projection': '3d'})
for n_clusters, ax in zip(range(2, 11), axes.flat):
    kmedoids = KMedoids(n_clusters=n_clusters)
    cluster_labels = kmedoids.fit_predict(X_scaled)
    cluster_centers = kmedoids.cluster_centers_
```

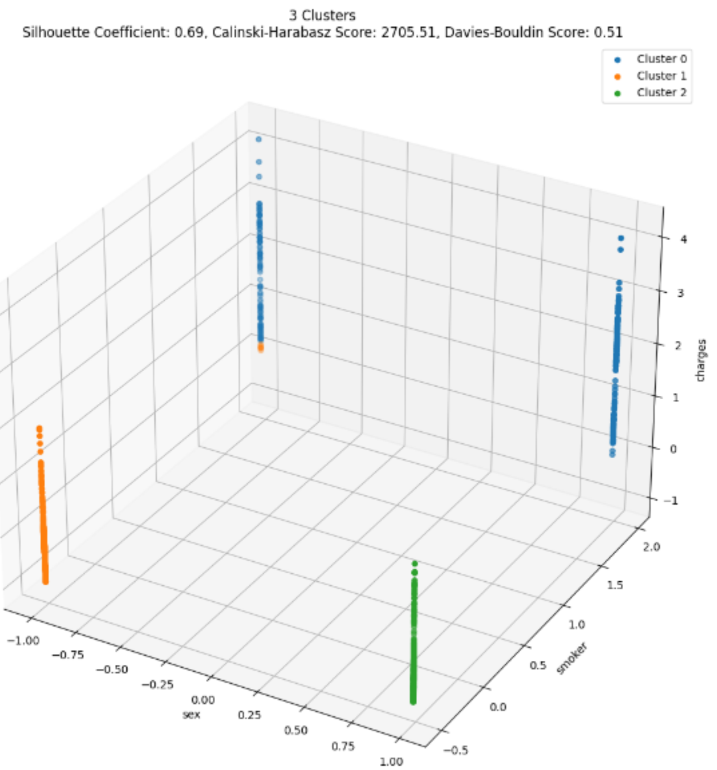
  - Marker:

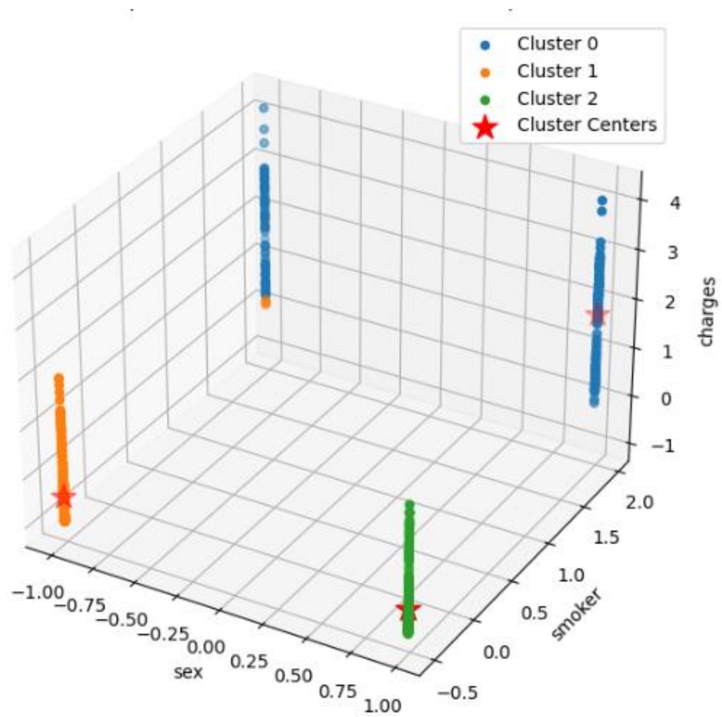
```
ax.scatter(cluster_centers[: 0], cluster_centers[:, 1],
cluster_centers[:, 2],
              marker='*', color='red', s=200, label='Cluster Centers')
```

- Visualizing 10 Clusters:



- Most optimal cluster





- Centroids of cluster 3:

```
Cluster 3 Center Coordinates:
         sex    smoker   charges
8   0.989591 -0.507463 -0.727539
12  0.989591  1.970587  1.780865
17  0.989591 -0.507463 -0.061692
23  0.989591 -0.507463 -0.061692
30  0.989591 -0.507463 -0.061692
38  0.989591 -0.507463 -0.061692
47  0.989591 -0.507463 -0.061692
```

- Coordinates of 3 clusters:

Cluster 0 points:

	sex	smoker	charges
0	-1.010519	-0.507463	-0.753434
1	-1.010519	-0.507463	-0.708986
2	-1.010519	-0.507463	-0.687347
3	-1.010519	-0.507463	-0.574670
4	-1.010519	-0.507463	-0.601455
...	...	...	...
116	-1.010519	-0.507463	-0.736309
117	-1.010519	-0.507463	-0.750181
118	-1.010519	-0.507463	-0.644169
119	-1.010519	-0.507463	-0.737681
120	-1.010519	-0.507463	-0.718601

121 rows × 3 columns

Cluster 1 points:

	sex	smoker	charges
0	0.989591	-0.507463	-0.567017
1	0.989591	-0.507463	-0.583745
2	0.989591	-0.507463	-0.578081
3	0.989591	-0.507463	-0.594008
4	0.989591	-0.507463	-0.385297
...	...	...	...
167	0.989591	-0.507463	-0.661597
168	0.989591	-0.507463	-0.528228
169	0.989591	-0.507463	-0.522863
170	0.989591	-0.507463	-0.648086
171	0.989591	-0.507463	-0.321550

172 rows × 3 columns

Cluster 2 points:

	sex	smoker	charges
0	-1.010519	-0.507463	-0.785908
1	-1.010519	-0.507463	-0.914646
2	-1.010519	-0.507463	-0.844606
3	-1.010519	-0.507463	-0.815917
4	-1.010519	-0.507463	-0.802405
...	...	...	...
115	-1.010519	-0.507463	-0.892637
116	-1.010519	-0.507463	-0.953183
117	-1.010519	-0.507463	-0.914002
118	-1.010519	-0.507463	-0.961596
119	-1.010519	-0.507463	-0.930362

120 rows × 3 columns

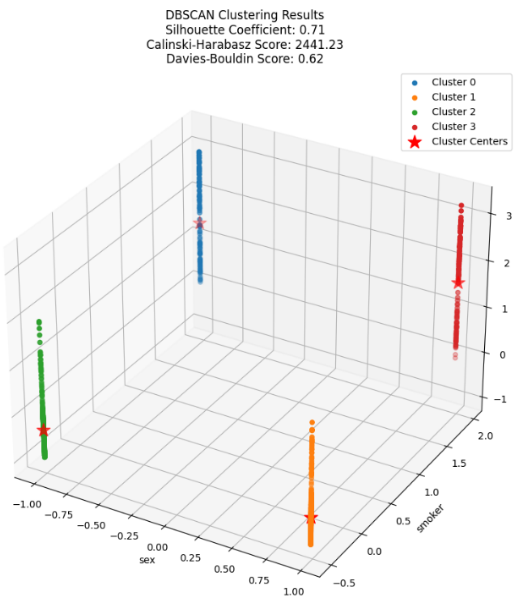
4. DB Scan Clustering + Visualizations + Clustering Quality:

- Intializng the model:

```
dbscan = DBSCAN()
cluster_labels_dbscan = dbscan.fit_predict(X_scaled)
unique_clusters = set(cluster_labels_dbscan[cluster_labels_dbscan != -
1])
```

- Visulaizing the model:

  - Best Clustering result



  - Coordinates

Cluster Centers Coordinates:

	sex	smoker	charges
0	-1.010519	1.970587	1.375056
1	0.989591	-0.507463	-0.428171
2	-1.010519	-0.507463	-0.372403
3	0.989591	1.970587	1.603531

Cluster 2:

	sex	smoker	charges
0	-1.010519	-0.507463	-0.785908
1	-1.010519	-0.507463	-0.415500
2	-1.010519	-0.507463	-0.494728
3	-1.010519	-0.507463	1.293027
4	-1.010519	-0.507463	-0.180059
...	...	...	...
542	-1.010519	-0.507463	-0.204410
543	-1.010519	-0.507463	-0.153545
544	-1.010519	-0.507463	-0.914002
545	-1.010519	-0.507463	-0.961596
546	-1.010519	-0.507463	-0.930382

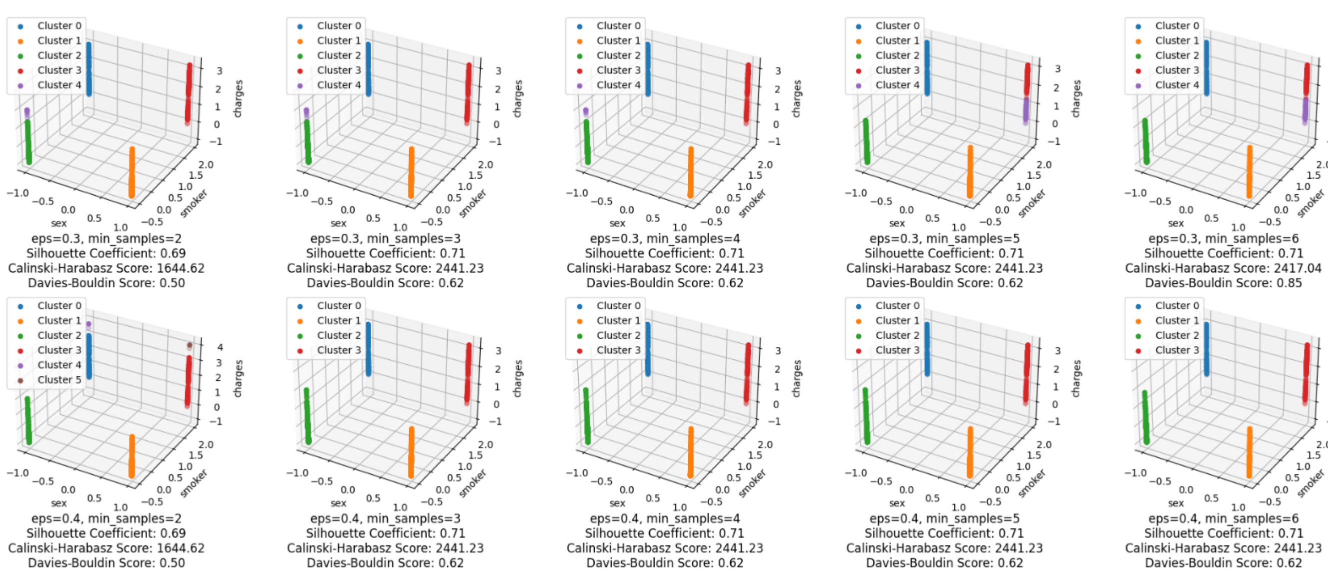
547 rows × 3 columns

Cluster 3:

	sex	smoker	charges
0	0.989591	1.970587	2.175983
1	0.989591	1.970587	1.946807
2	0.989591	1.970587	2.101574
3	0.989591	1.970587	1.843392
4	0.989591	1.970587	3.132806
...	...	...	...
152	0.989591	1.970587	2.763024
153	0.989591	1.970587	2.028775
154	0.989591	1.970587	0.659945
155	0.989591	1.970587	0.677549
156	0.989591	1.970587	1.225140

157 rows × 3 columns

  - Overall clusters test



5. Conclusion:

Metrics	K-Medoid	DB Scan
Silhouette	0.69	0.71
Calinski-Harabasz	2705.51	2441.23
Davies-Bouldin	0.51	0.62

Overall, DBSCAN demonstrates superior cluster separation for the selected dataset according to Silhouette measure. The features 'sex', 'smoking', and 'charges' were selected due to their optimal inertia, suggesting they play a key role in determining individual insurance.

Cooked by:

1-	Menna Elminshawy	20217011
2-	Jana Mohamed Nayef	20216129
3-	Radwa Belal	20217005
4-	Mariam Behairy	20216094
5-	Ahmad Wael Abdelaziz	20216016
6-	Yassin Ehab	20216117