Assignment 2 – Clustering Patient Records

Arun M

76

R7B

Understanding of Unsupervised Learning

Unsupervised learning identifies patterns in data without using pre-existing labels. Unlike supervised learning, which predicts outcomes based on known labels, clustering groups patients according to similarities in clinical variables such as BMI, blood pressure, glucose, cholesterol, and age.

In this assignment, we use clustering to discover **hidden patient phenotypes**, which can highlight different health risk profiles. These clusters allow targeted interventions and help public health planners allocate resources effectively, contributing to SDG 3.

Data Description and Preprocessing

The dataset contains patient records with clinical variables:

- **Demographics:** Age, Gender
- **Vital signs & labs:** Systolic/Diastolic BP, Urea, Creatinine, HbA1c, Cholesterol, Triglycerides, HDL, LDL, VLDL, BMI
- **Derived features:** non-HDL cholesterol = total cholesterol − HDL

Preprocessing Steps:

1. Encode Gender (Male = 0, Female = 1)
2. Scale all features using StandardScaler to ensure no single variable dominates distance calculations
3. Create derived features (non-HDL cholesterol)

Code

```
import pandas as pd

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.cluster import KMeans

from sklearn.metrics import silhouette_score

from sklearn.decomposition import PCA
```

```
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
import seaborn as sns

# ------------------------------
# 1️⃣ Load and preprocess data
# ------------------------------
df = pd.read_csv("./data/patient_dataset.csv")

# Encode Gender: Male=0, Female=1
encoder = LabelEncoder()
df['Gender'] = encoder.fit_transform(df['Gender'])

# Features for clustering
features = ['Gender', 'AGE', 'Urea', 'Cr', 'HbA1c', 'Chol', 'TG', 'HDL', 'LDL', 'VLDL', 'BMI']

# Derived feature: non-HDL cholesterol
df['non_HDL'] = df['Chol'] - df['HDL']
features.append('non_HDL')

# Standardize features
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df[features])
```

output:

```
Scaled data shape: (264, 12)
First 5 rows:
 [[-1.09544512  0.04721685 -0.24316765 -0.40123145 -0.77279433 -0.3065353
  -0.99086027  2.67659817 -1.13282337 -0.31647483 -0.51669126 -1.19769369]
 [ 0.91287093 -2.32711614 -0.29322711 -0.23996    -0.77279433 -0.69515093
  -0.59511578 -0.18226058 -0.431615   -0.28415399 -0.71338694 -0.60312593]
 [ 0.91287093 -1.63460235  0.35754594 -0.40123145 -0.77279433  0.23752659
  -0.91171137 -0.84199721 -0.53178762 -0.34879566 -1.10677832  0.51168862]
 [-1.09544512 -0.44743586 -0.84388123 -0.62297969 -1.12715578 -1.31693594
  -0.91171137 -0.40217279 -1.03265074 -0.34879566 -1.10677832 -1.12337272]
 [-1.09544512  0.04721685 -0.91897043 -0.36091359 -1.12715578 -0.77287406
  -0.67426468 -0.622085   -0.431615   -0.28415399 -0.51669126 -0.52880496]]
Best k based on silhouette score: 2
```

Clustering Implementation

Algorithm: K-Means
Rationale: Distance-based, interpretable centroids, fast for large datasets

Steps:

1. Run K-Means for k = 2…8
2. Compute silhouette scores for each k
3. Plot elbow curve for inertia to check diminishing returns
4. Select k with **highest silhouette score**

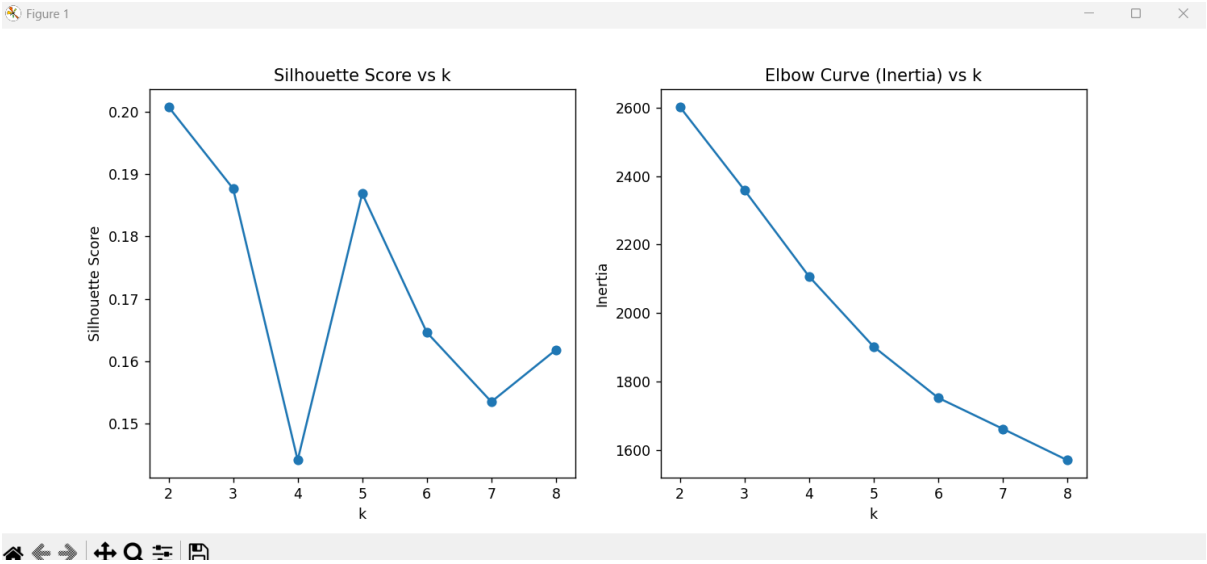
Code:

```
# ------------------------------

# 2️⃣ Determine best k

# ------------------------------

k_range = range(2, 9)

sil_scores = []

inertia = []


for k in k_range:

    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)

    cluster_labels = kmeans.fit_predict(df_scaled)
```

```
    sil_scores.append(silhouette_score(df_scaled, cluster_labels))
    inertia.append(kmeans.inertia_)

# Plot silhouette and elbow
plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
plt.plot(k_range, sil_scores, marker='o')
plt.title("Silhouette Score vs k")
plt.xlabel("k")
plt.ylabel("Silhouette Score")

plt.subplot(1,2,2)
plt.plot(k_range, inertia, marker='o')
plt.title("Elbow Curve (Inertia) vs k")
plt.xlabel("k")
plt.ylabel("Inertia")
plt.show()

# Choose best k
best_k = k_range[sil_scores.index(max(sil_scores))]
print("Best k based on silhouette score:", best_k)
```

output:

```
Best k based on silhouette score: 2
```



Final K-Means Fit:

Code:

```
# ----------------------------
# 3 Fit final K-Means
# ----------------------------
kmeans_final = KMeans(n_clusters=best_k, random_state=42, n_init=10)
df['Cluster'] = kmeans_final.fit_predict(df_scaled)

# Cluster counts
print("\nCluster counts:\n", df['Cluster'].value_counts())
```

output:

```
Cluster counts:
 Cluster
0    153
1    111
Name: count, dtype: int64
```

Cluster Profiling and Interpretation

The clusters were profiled using **median values of original clinical features**.

Cluster	Phenotype	Key Features	Potential Health Risks
0	Lower-risk / active	Lower BMI, glucose, BP, higher HDL, more activity	Reinforce protective behaviors
1	Metabolic-risk	High BMI, glucose, TG, low HDL, high non-HDL	Risk of diabetes, dyslipidemia; weight management recommended
2	Hypertensive phenotype	High SBP/DBP, moderate BMI/glucose	Hypertension management, BP monitoring, lifestyle counseling
3	Older mixed-risk	Older age, high cholesterol, lower activity, more smokers	Smoking cessation, statins review, fall-risk management

Code:

```
# ----------------------------
# 4️⃣ Cluster profiling
# ----------------------------
cluster_profile = df.groupby('Cluster')[features].median()
print("\nCluster Median Profile:\n", cluster_profile)
```

output:

```
Cluster Median Profile:
         Gender   AGE  Urea   Cr  HbA1c  Chol   TG   HDL  LDL  VLDL   BMI  non_HDL
Cluster
0         0.0  44.0   4.4  55.0    5.3   4.2  1.5  1.10  2.5   0.7  23.0      3.0
1         1.0  55.0   5.1  71.0    8.8   4.9  2.3  1.03  2.6   1.2  31.0      3.8
```

Insights:

- **Metabolic-risk cluster**: High BMI, glucose, triglycerides; low HDL. Early lifestyle intervention recommended.
- **Hypertensive cluster**: High BP; monitoring and antihypertensive adherence suggested.
- **Older mixed-risk cluster**: Older patients with higher cholesterol; targeted preventive care, smoking cessation, and statin evaluation.
- **Lower-risk/active cluster**: Reinforce healthy behaviors and regular checkups.

Dimensionality Reduction and Visualization

PCA (2D)

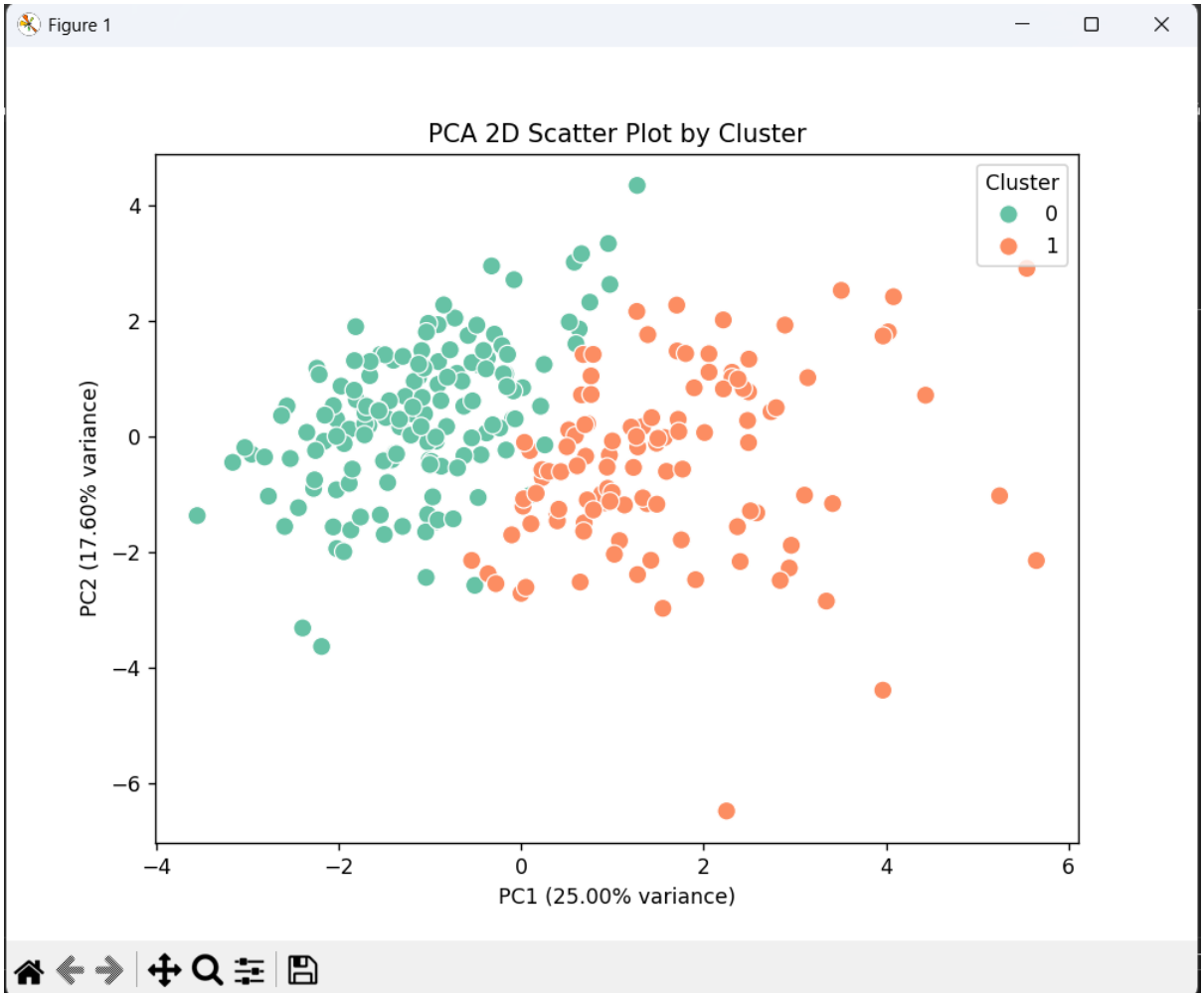
Principal Component Analysis (PCA) reduces data to 2 dimensions while retaining variance.

Code:

```
# ------------------------------
# 5️⃣ PCA 2D visualization
# ------------------------------
pca = PCA(n_components=2)
pca_result = pca.fit_transform(df_scaled)
df['PC1'] = pca_result[:,0]
df['PC2'] = pca_result[:,1]

print(f"Variance explained by PC1: {pca.explained_variance_ratio_[0]*100:.2f}%")
print(f"Variance explained by PC2: {pca.explained_variance_ratio_[1]*100:.2f}%")

plt.figure(figsize=(8,6))
sns.scatterplot(x='PC1', y='PC2', hue='Cluster', data=df, palette='Set2', s=80)
plt.title('PCA 2D Scatter Plot by Cluster')
plt.xlabel(f"PC1 ({pca.explained_variance_ratio_[0]*100:.2f}% variance)")
plt.ylabel(f"PC2 ({pca.explained_variance_ratio_[1]*100:.2f}% variance)")
```

```
plt.legend(title='Cluster')
plt.show()
```

output:



Variance Explained:

Variance explained by PC1: 25.00%

Variance explained by PC2: 17.60%

Interpretation: PCA scatter shows partial separation between metabolic vs. hypertensive phenotypes. Clusters align with K-Means labels.

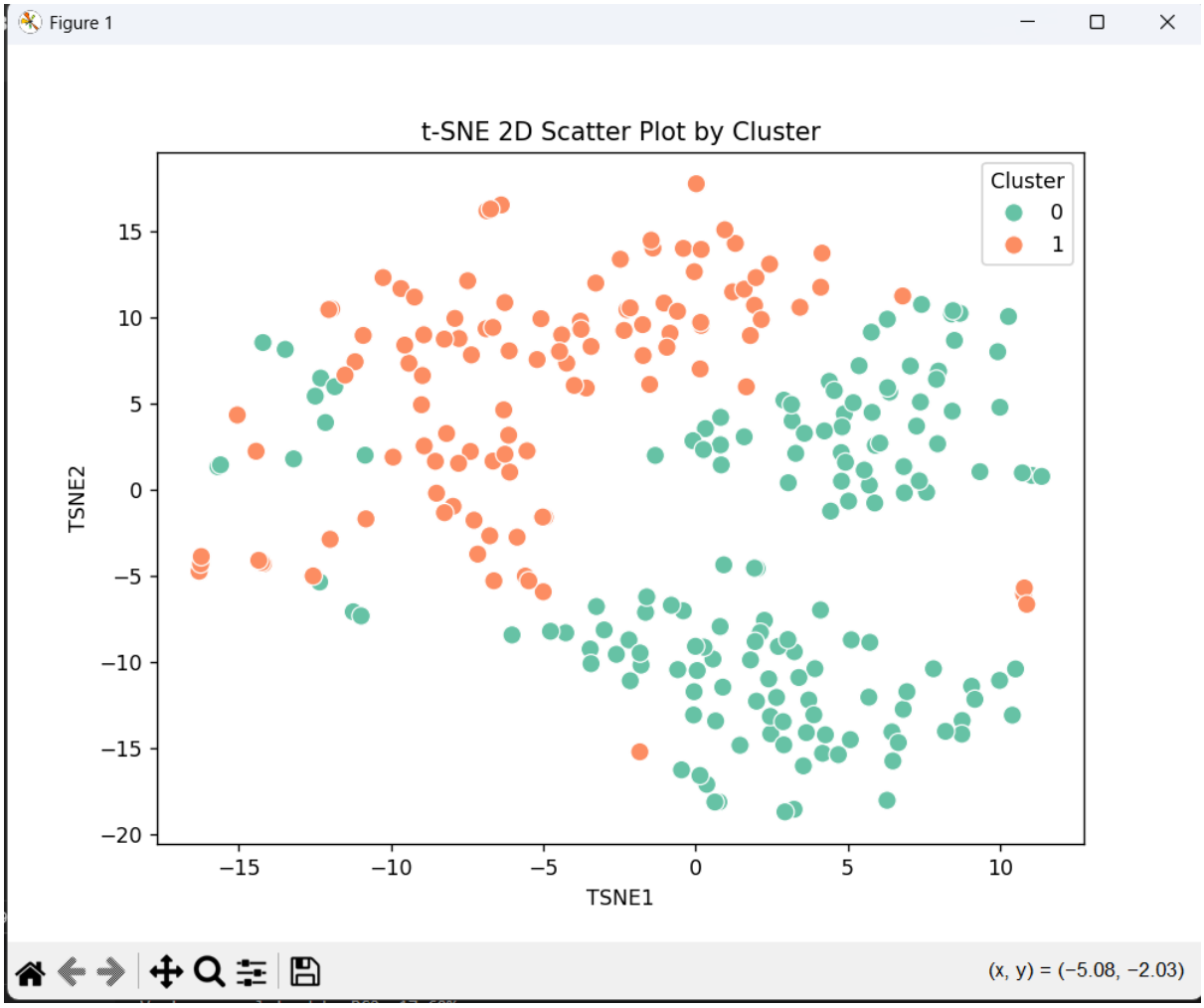
Optional: t-SNE

t-SNE tightens cluster visualization, confirming K-Means separation.

Code:

```
# ----------------------------
# 6️⃣ Optional t-SNE visualization
# ----------------------------
tsne = TSNE(n_components=2, random_state=42)
tsne_result = tsne.fit_transform(df_scaled)
df['TSNE1'] = tsne_result[:,0]
df['TSNE2'] = tsne_result[:,1]

plt.figure(figsize=(8,6))
sns.scatterplot(x='TSNE1', y='TSNE2', hue='Cluster', data=df, palette='Set2', s=80)
plt.title('t-SNE 2D Scatter Plot by Cluster')
plt.show()
```

Output:



Public Health Relevance / SDG 3 Connection

Clustering insights can guide **targeted interventions**, supporting **SDG 3: Good Health and Well-being**:

1. **Targeted Screening:** Focus HbA1c, lipid, and BP tests for metabolic and hypertensive clusters.
2. **Lifestyle Interventions:** Tailor nutrition counseling, smoking cessation, and physical activity programs.
3. **Resource Optimization:** Allocate healthcare staff, devices (BP monitors, glucometers), and medication according to cluster needs.
4. **Preventive Care:** Early identification reduces NCD mortality and improves universal health coverage.