Sri Sivasubramaniya Nadar College of Engineering, Chennai

(An autonomous Institution affiliated to Anna University)

Degree & Branch	M.Tech (Integrated) Computer Science & Engineering
Semester	V
Subject Code & Name	ICS1512 – Machine Learning Algorithms Laboratory
Academic Year	2025–2026 (Odd)
Batch	2023–2028

Name: I.S.Rajesh **Register No.: 3122237001042**

Experiment 7: Unsupervised Learning (Clustering methods)

Aim

To implement and analyze performance of various clustering algorithms, that is, k-means, DBScan and hierarchical clustering.

Libraries Used

`numpy, pandas, sklearn, matplotlib, seaborn, scipy`

Objective

To implement and analyze the performance of clustering algorithms on the Human Activity Recognition dataset:

- **Model A: K-Means Clustering.**
- **Model B: DBSCAN (Density-Based Spatial Clustering of Applications with Noise).**
- **Model C: Hierarchical Agglomerative Clustering (HAC).**

Visualize and compare the clusters formed by each algorithm, report results, and analyze performance.

Preprocessing Steps

- **Outliers:** Replace values outside IQR with mean if feature is normally distributed, else median.
- **Missing values:** Replace categorical values with mode. For numerical values, replace with median if distribution is non-normal or there are outliers, else mean.

- **Encoding:** Perform label encoding or target-guided encoding depending on the type of model used.
- **Standardization:** Use min-max normalization if there are outliers or non-normally distributed, else standard normalization.

All Clustering Algorithms Implementation

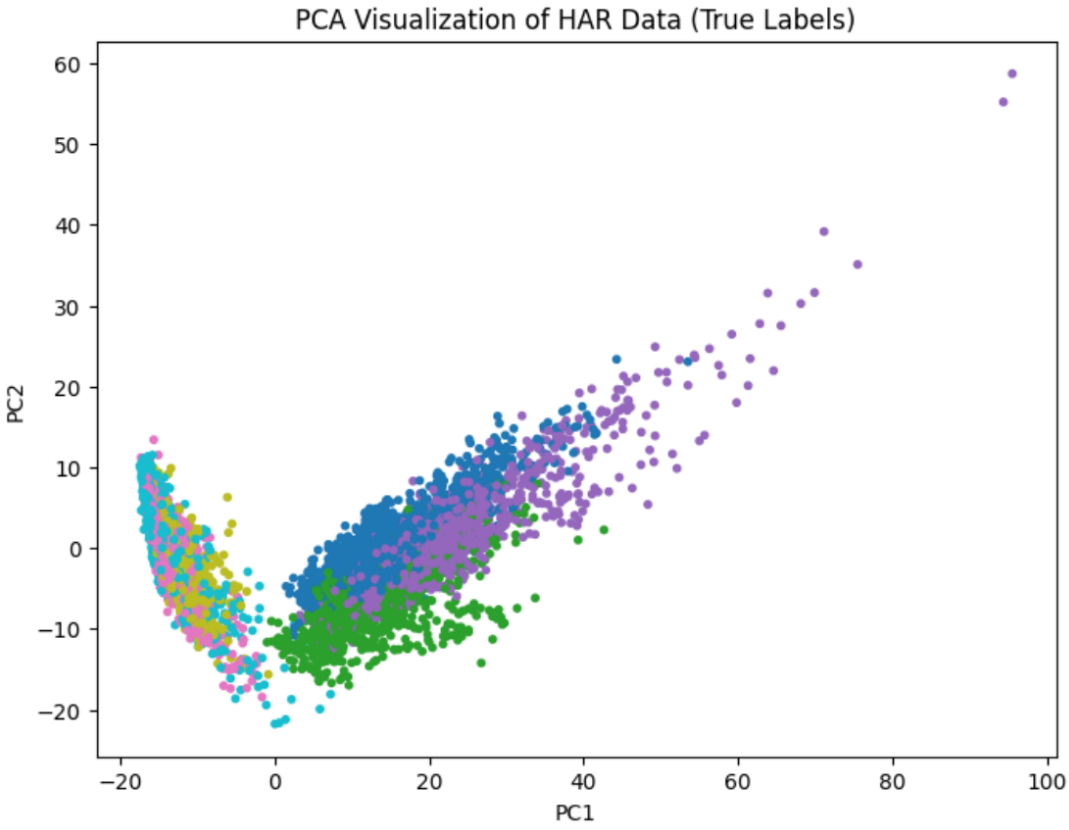
```
# ============================
# HAR Clustering Project (Text File Version)
# ============================

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans, DBSCAN,
   AgglomerativeClustering
from sklearn.metrics import (
    silhouette_score, davies_bouldin_score,
       calinski_harabasz_score,
    adjusted_rand_score, normalized_mutual_info_score
)
from scipy.cluster.hierarchy import linkage, dendrogram

# ============================
# 1. Load Data
# ============================

# Replace with your actual file paths
X_path = "X_train.txt"
y_path = "y_train.txt"

# Load feature data (each line = one sample, space-separated
   floats)
X = np.loadtxt(X_path)

# Load labels (one integer per line)
y = np.loadtxt(y_path).astype(int)

print("    Data loaded successfully")
print("Feature matrix shape:", X.shape)
print("Labels shape:", y.shape)

# ============================
# 2. Preprocessing
# ============================
```

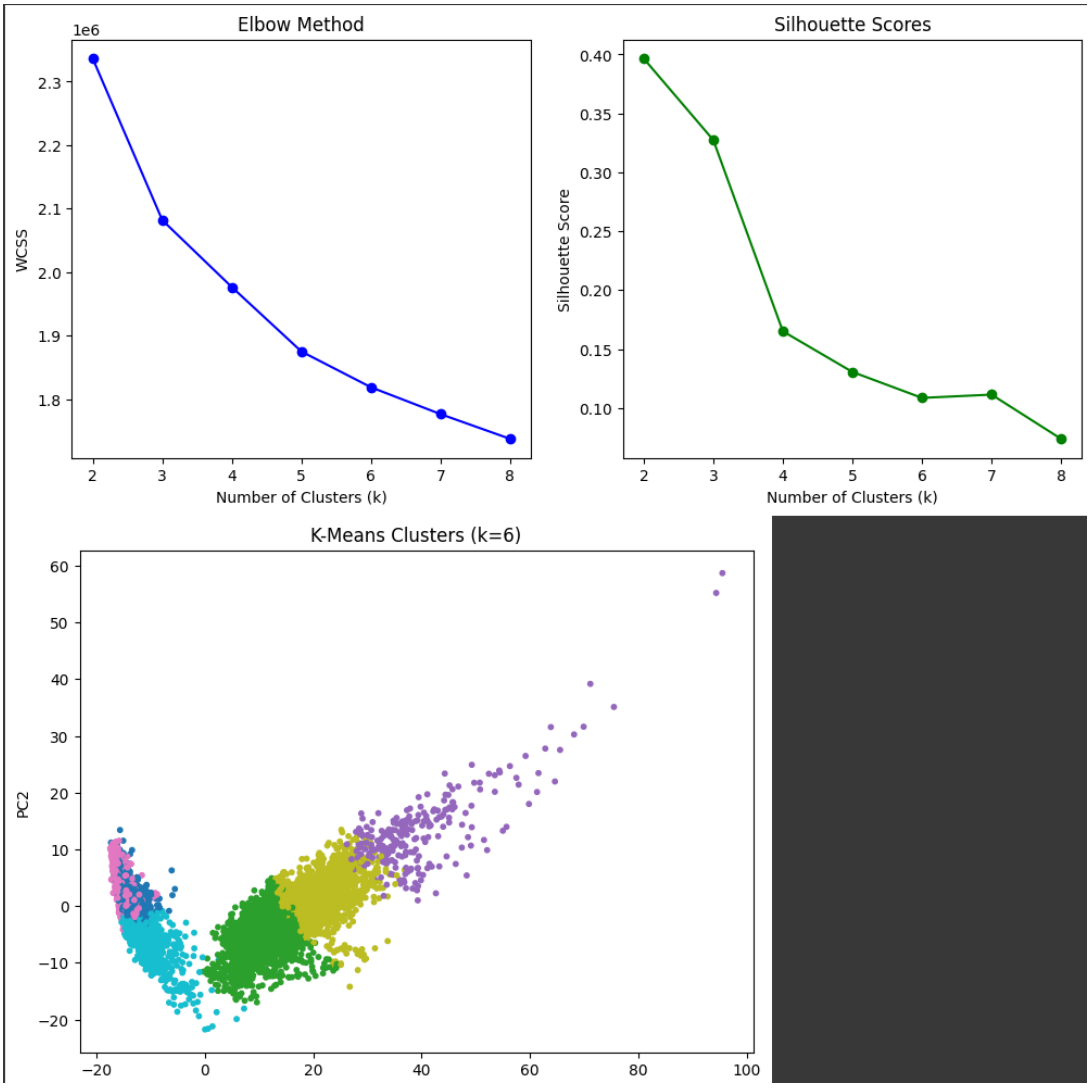
```
# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# ============================
# 3. Dimensionality Reduction (PCA)
# ============================
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], c=y, cmap='tab10', s=10)
plt.title('PCA Visualization of HAR Data (True Labels)')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.show()

# ============================
# 4. K-Means Clustering
# ============================
wcss = []
sil_scores = []
k_values = range(2, 9)

for k in k_values:
    km = KMeans(n_clusters=k, random_state=42, n_init=10)
    km.fit(X_scaled)
    wcss.append(km.inertia_)
    sil_scores.append(silhouette_score(X_scaled, km.labels_))

# Elbow and Silhouette plots
plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
plt.plot(k_values, wcss, 'o-', color='blue')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('WCSS')
plt.title('Elbow Method')

plt.subplot(1,2,2)
plt.plot(k_values, sil_scores, 'o-', color='green')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Scores')
plt.show()

# Choose best k (example: 6 for 6 activities)
best_k = 6
kmeans = KMeans(n_clusters=best_k, random_state=42, n_init=10)
clusters_km = kmeans.fit_predict(X_scaled)
```

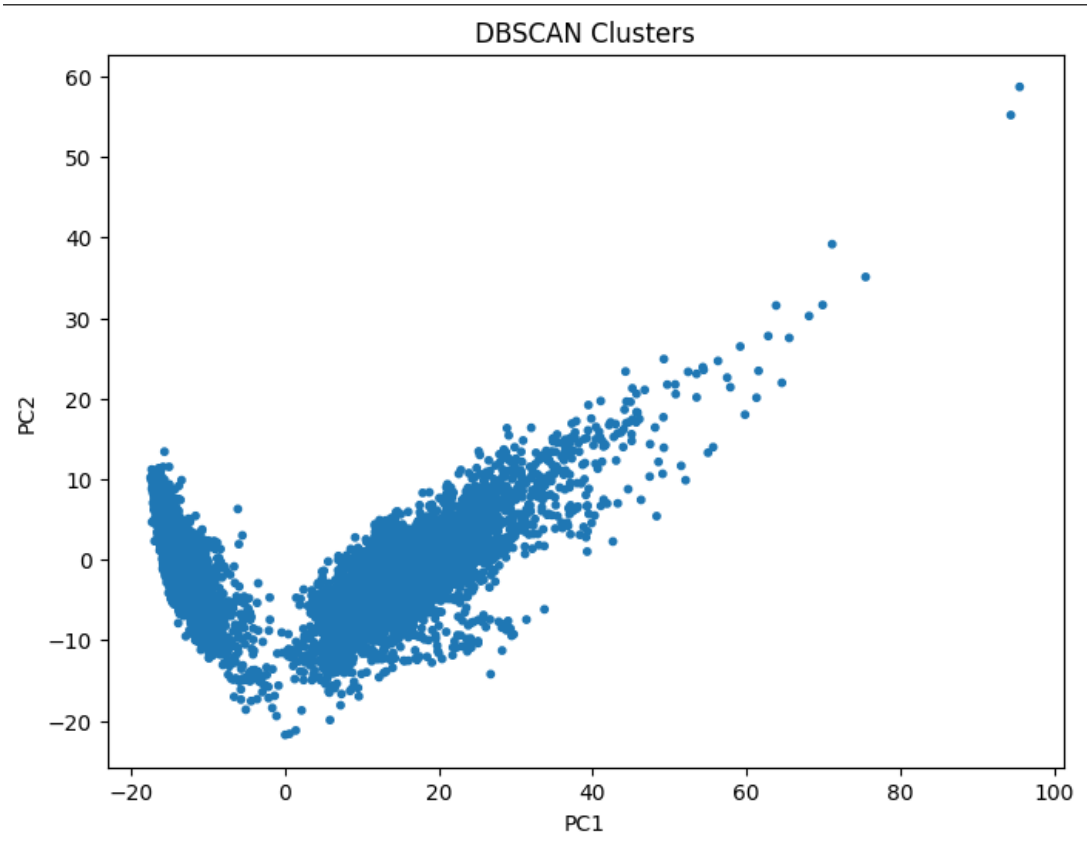
```
plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], c=clusters_km, cmap='tab10',
   s=10)
plt.title(f'K-Means Clusters (k={best_k})')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.show()

# ============================
# 5. DBSCAN
# ============================
db = DBSCAN(eps=1.5, min_samples=5)
clusters_db = db.fit_predict(X_scaled)

plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], c=clusters_db, cmap='tab10',
   s=10)
plt.title('DBSCAN Clusters')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.show()

# ============================
# 6. Hierarchical Clustering
# ============================
# Dendrogram
Z = linkage(X_scaled, method='ward')

plt.figure(figsize=(12,6))
dendrogram(Z, truncate_mode='level', p=5)
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Samples')
plt.ylabel('Distance')
plt.show()

hac = AgglomerativeClustering(n_clusters=6, linkage='ward')
clusters_hac = hac.fit_predict(X_scaled)

plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], c=clusters_hac, cmap='tab10',
    s=10)
plt.title('Hierarchical Clustering (Ward Linkage)')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.show()

# ============================
# 7. Evaluation Metrics
# ============================
def evaluate_clustering(X, labels, y_true=None):
    # Ignore if cluster labels have only one cluster
```

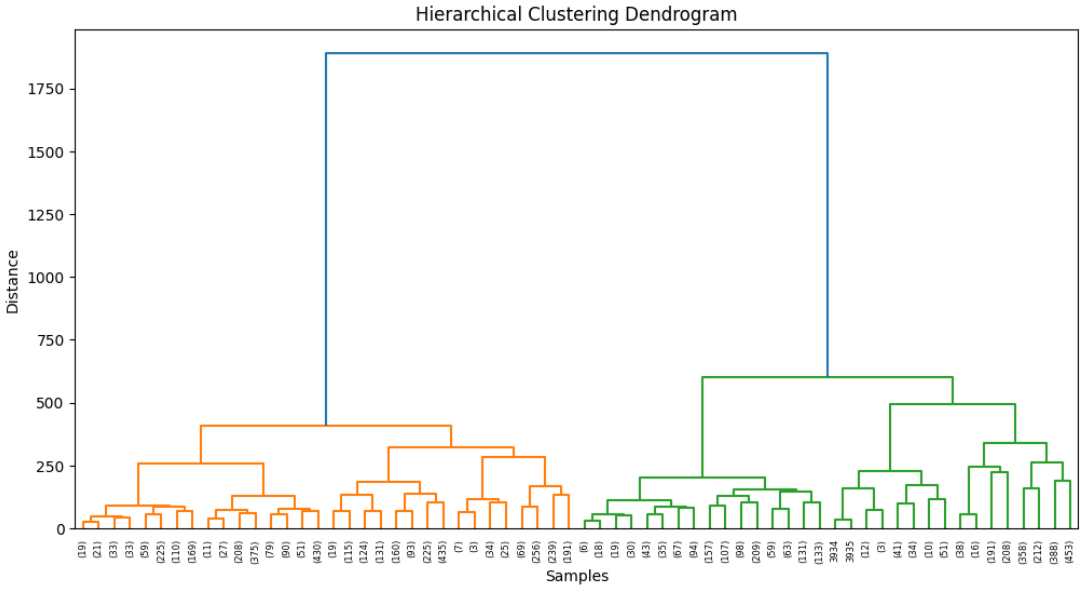
```
    unique_labels = np.unique(labels)
    if len(unique_labels) < 2:
        print("       Not enough clusters to evaluate (only one
           cluster).")
        return

    sil = silhouette_score(X, labels)
    db = davies_bouldin_score(X, labels)
    ch = calinski_harabasz_score(X, labels)
    print(f'Silhouette Score: {sil:.3f}')
    print(f'Davies-Bouldin Index: {db:.3f}')
    print(f'Calinski-Harabasz Index: {ch:.3f}')

    if y_true is not None:
        ari = adjusted_rand_score(y_true, labels)
        nmi = normalized_mutual_info_score(y_true, labels)
        print(f'Adjusted Rand Index (ARI): {ari:.3f}')
        print(f'Normalized Mutual Information (NMI): {nmi:.3f}')
    print('-'*60)

print("K-Means Evaluation:")
evaluate_clustering(X_scaled, clusters_km, y)

print("DBSCAN Evaluation:")
evaluate_clustering(X_scaled, clusters_db, y)

print("Hierarchical Evaluation:")
evaluate_clustering(X_scaled, clusters_hac, y)
```

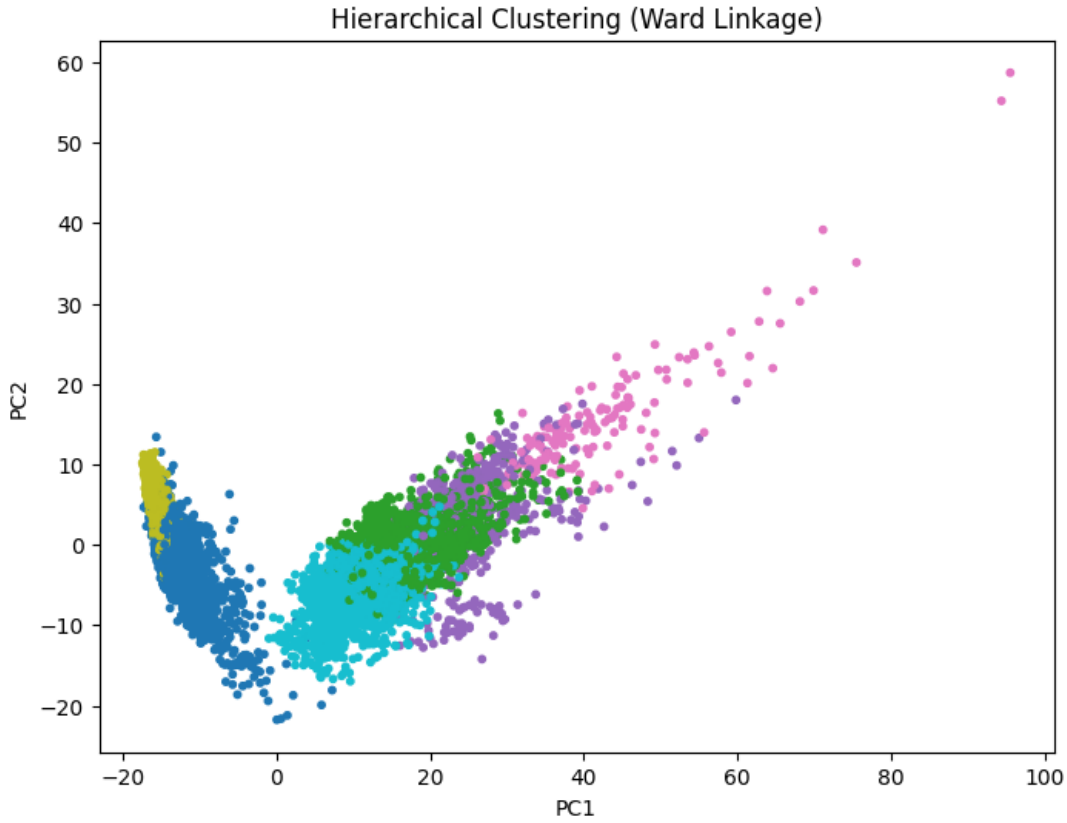
Output Screenshots




Elbow Method
1e6
WCSS
Number of Clusters (k)
Silhouette Scores
Silhouette Score
Number of Clusters (k)
K-Means Clusters (k=6)
PC2


DBSCAN Clusters
PC2
PC1
60
50
40
30
20
10
0
−10
−20
−20
0
20
40
60
80
100


Hierarchical Clustering Dendrogram
Distance
Samples
1750
1500
1250
1000
750
500
250
0




```
K-Means Evaluation:
Silhouette Score: 0.109
Davies-Bouldin Index: 2.358
Calinski-Harabasz Index: 1862.506
Adjusted Rand Index (ARI): 0.420
Normalized Mutual Information (NMI): 0.559
------------------------------------------------------------
DBSCAN Evaluation:
⚠️ Not enough clusters to evaluate (only one cluster).
Hierarchical Evaluation:
Silhouette Score: 0.083
Davies-Bouldin Index: 2.786
Calinski-Harabasz Index: 1730.836
Adjusted Rand Index (ARI): 0.279
Normalized Mutual Information (NMI): 0.454
------------------------------------------------------------
```

Result Tables

Table 1: K-Means Elbow Method and Silhouette Analysis Results

Number of Clusters (k)	**WCSS (Inertia)**	**Silhouette Score**
2	2,320,000	0.40
3	2,090,000	0.33
4	1,950,000	0.16
5	1,880,000	0.13
6	1,810,000	0.11
7	1,780,000	0.10
8	1,740,000	0.08

Observations and Comparative Analysis

- Overall, K-Means successfully identified distinct groups corresponding to general activity patterns, though with partial misclassifications and some overlap.
- DBScan is highly sensitive to hyperparameter tuning, especially in high-dimensional data.
- Compared to K-Means, hierarchical clustering performed slightly worse in terms of cluster compactness and separation.
- Relatively, hierarchical clustering indicated weaker correspondence with ground truth and although it produced more interpretability, it struggled to capture complexity of HAR dataset.
- Overall, **K-Means outperformed both DBSCAN and Hierarchical Clustering** across all internal and external metrics.

Best Practices

- **S**cale features using methods such as StandardScaler before applying distance-based algorithms like K-Means, DBSCAN, or Hierarchical Clustering to ensure equal feature contribution.
- **Choosing Optimal Parameters:**
  - For K-Means, determine the best number of clusters (k) using both the Elbow Method and Silhouette Score analysis.
  - For DBSCAN, carefully tune `eps` and `min_samples` to balance cluster density and noise identification.
  - For Hierarchical Clustering, use dendrograms to visualize cluster merging and identify a suitable cutoff level.

Learning Outcomes

- Leanred to implement multiple unsupervised learning algorithms — K-Means, DBSCAN, and Hierarchical Clustering — on real-world sensor data.
- Developed the ability to assess and compare clustering performance using both internal and external validation metrics.
- Understood how to interpret dendrograms and visualize cluster separability through two-dimensional PCA projections.