```python
# ============================================================
# Foundations of Data Science Project
# Dataset: Cleaned_Matches_Dataset.csv
# Topic: Clustering (Distance Metrics, KMeans, Hierarchical)
# ============================================================

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans, AgglomerativeClustering
from sklearn.decomposition import PCA
from scipy.spatial.distance import cdist, pdist, squareform

# ================================
# 1. Load Dataset
# ================================
df = pd.read_csv("Cleaned_Matches_Dataset.csv")

# Drop unwanted index column if present
if "Unnamed: 0" in df.columns:
    df = df.drop(columns=["Unnamed: 0"])

print("Shape of dataset:", df.shape)

# Select numeric features for clustering
num_cols = ["result_margin", "target_runs", "target_overs"]
df_cluster = df[num_cols].fillna(0)

# Standardize
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_cluster)
```

```python
# ============================================================
# 2. Distance Metrics
# ============================================================
print("\n=== Distance Metrics ===")

euclidean_dist = cdist(X_scaled, X_scaled, metric="euclidean")
manhattan_dist = cdist(X_scaled, X_scaled, metric="cityblock")
cosine_dist = cdist(X_scaled, X_scaled, metric="cosine")

print("Euclidean Distance (first 5x5 block):\n", euclidean_dist[:5,:5])
print("Manhattan Distance (first 5x5 block):\n", manhattan_dist[:5,:5])
print("Cosine Distance (first 5x5 block):\n", cosine_dist[:5,:5])

# ============================================================
# 3. K-Means Clustering (Lloyd's Algorithm)
# ============================================================
print("\n=== K-Means Clustering (Lloyd's Algorithm) ===")

k = 3
kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
labels_kmeans = kmeans.fit_predict(X_scaled)

print("K-Means Cluster Centers:\n", kmeans.cluster_centers_)

# PCA projection for visualization
pca = PCA(n_components=2, random_state=42)
proj = pca.fit_transform(X_scaled)

plt.scatter(proj[:,0], proj[:,1], c=labels_kmeans, cmap="viridis")
plt.title("K-Means Clustering (k=3)")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.show()
```

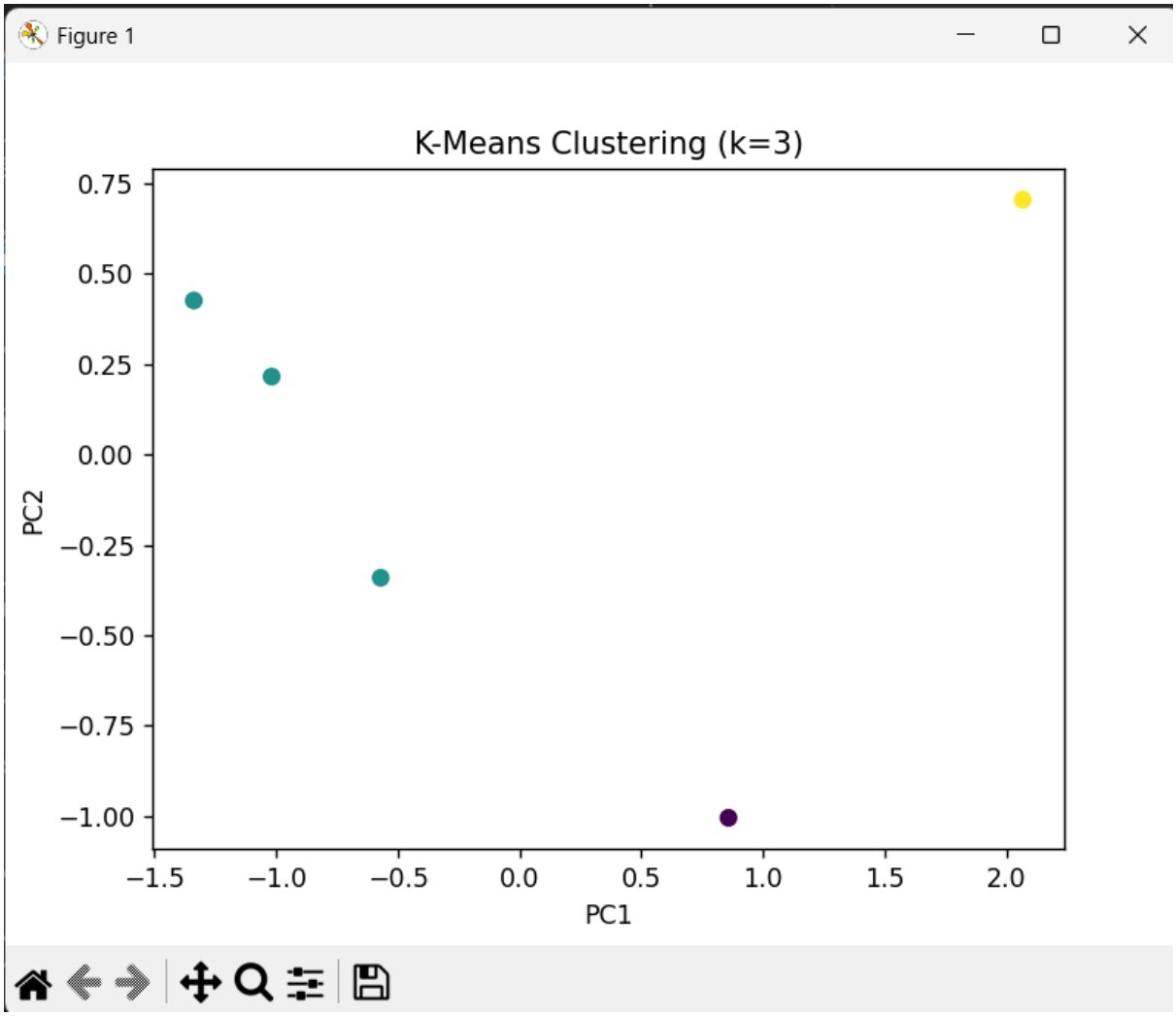
```python
# ============================================================
# 4. Hierarchical Agglomerative Clustering (HAC)
# ============================================================
print("\n=== Hierarchical Agglomerative Clustering (HAC) ===")

hac = AgglomerativeClustering(n_clusters=3, metric="euclidean", linkage="ward")
labels_hac = hac.fit_predict(X_scaled)

plt.scatter(proj[:,0], proj[:,1], c=labels_hac, cmap="plasma")
plt.title("Hierarchical Agglomerative Clustering (HAC)")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.show()

# ============================================================
# 5. Dendrogram (for Hierarchical Clustering)
# ============================================================
from scipy.cluster.hierarchy import dendrogram, linkage

Z = linkage(X_scaled, method="ward")  # ward = variance minimization
plt.figure(figsize=(10,5))
dendrogram(Z, truncate_mode="lastp", p=10, leaf_rotation=45, leaf_font_size=10, show_contra
plt.title("Hierarchical Clustering Dendrogram")
plt.xlabel("Cluster Size")
plt.ylabel("Distance")
plt.show()
```



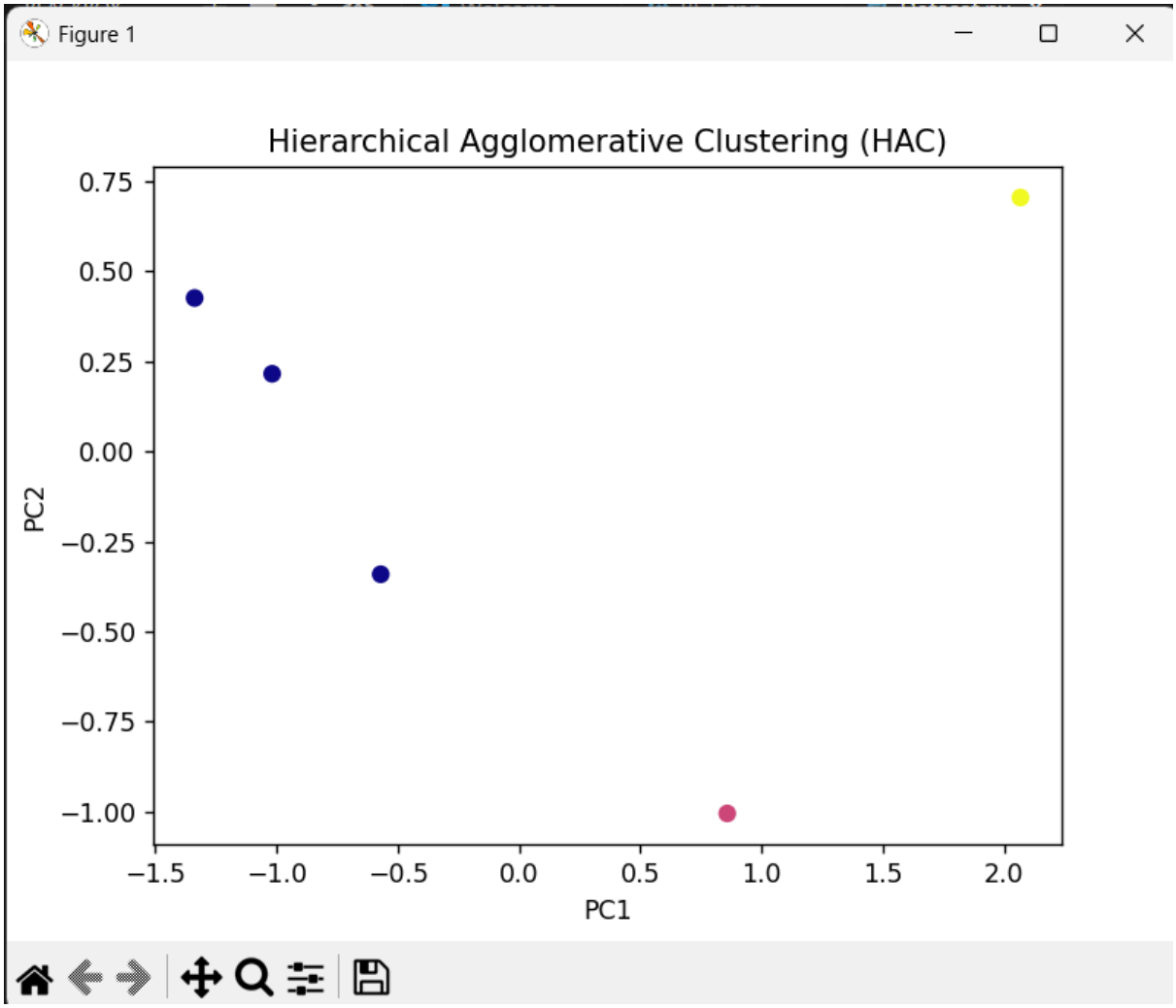


Figure 1
Hierarchical Agglomerative Clustering (HAC)
PC2
PC1
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

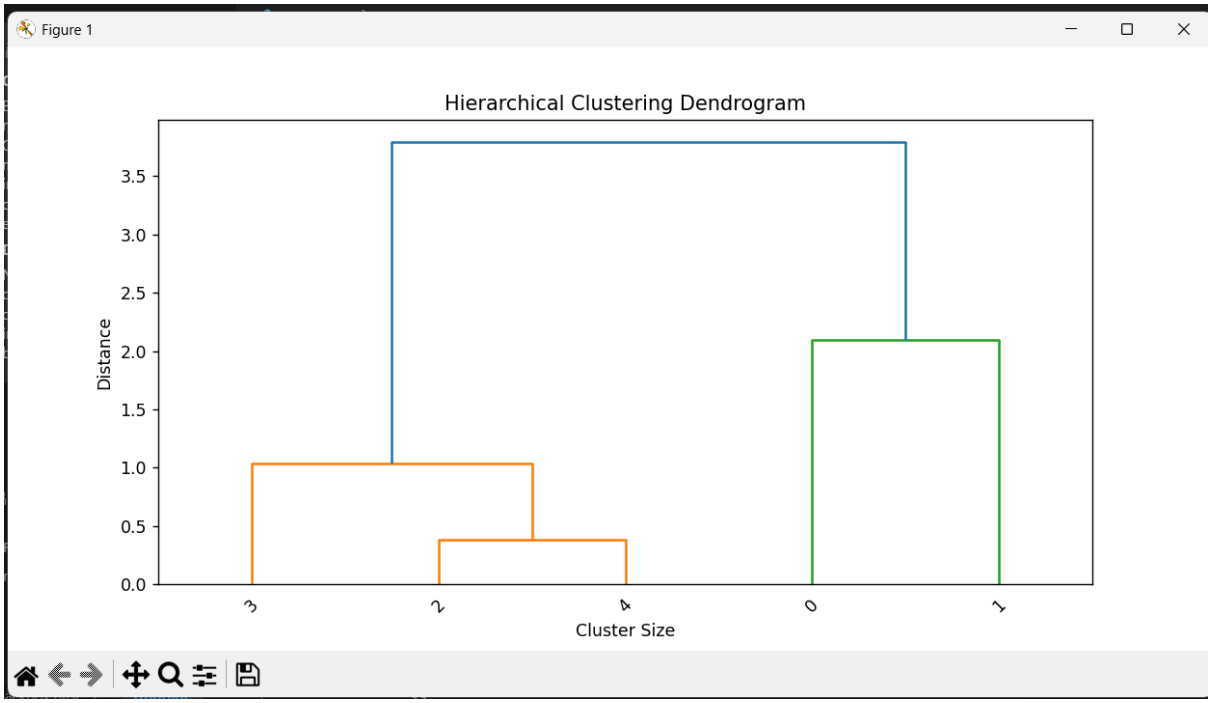


Figure 1
Hierarchical Clustering Dendrogram
Distance
Cluster Size
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
3
2
4
0
1