```python
# ============================================================
# Foundations of Data Science Project
# Dataset: Cleaned_Matches_Dataset.csv
# Topic: DBSCAN, Clustering Tendency & Quality
# ============================================================

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans, AgglomerativeClustering, DBSCAN
from sklearn.metrics import silhouette_score, davies_bouldin_score

# ================================
# 1. Load Dataset
# ================================
df = pd.read_csv("Cleaned_Matches_Dataset.csv")
if "Unnamed: 0" in df.columns:
    df = df.drop(columns=["Unnamed: 0"])

# Use numeric features
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)

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

```python
# ============================================================
# 2. K-Means (baseline)
# ============================================================
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
labels_kmeans = kmeans.fit_predict(X_scaled)

# ============================================================
# 3. Hierarchical Agglomerative Clustering (HAC)
# ============================================================
hac = AgglomerativeClustering(n_clusters=3, metric="euclidean", linkage="ward")
labels_hac = hac.fit_predict(X_scaled)

# ============================================================
# 4. DBSCAN
# ============================================================
print("\n=== DBSCAN Clustering ===")
dbscan = DBSCAN(eps=1.2, min_samples=2)  # tune eps based on dataset
labels_dbscan = dbscan.fit_predict(X_scaled)

print("DBSCAN Labels:", np.unique(labels_dbscan))

# Visualize DBSCAN clusters
plt.scatter(proj[:,0], proj[:,1], c=labels_dbscan, cmap="rainbow")
plt.title("DBSCAN Clustering")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.show()
```

```python
# ============================================================
# 5. Clustering Quality Metrics
# ============================================================
print("\n=== Clustering Quality ===")

def evaluate_clustering(X, labels, name):
    if len(set(labels)) > 1 and -1 not in set(labels):  # at least 2 clusters, no noise onl
        sil = silhouette_score(X, labels)
        dbi = davies_bouldin_score(X, labels)
        print(f"{name}: Silhouette={sil:.3f}, DBI={dbi:.3f}")
    else:
        print(f"{name}: Not enough valid clusters for metrics.")

evaluate_clustering(X_scaled, labels_kmeans, "K-Means")
evaluate_clustering(X_scaled, labels_hac, "HAC")
evaluate_clustering(X_scaled, labels_dbscan, "DBSCAN")

```

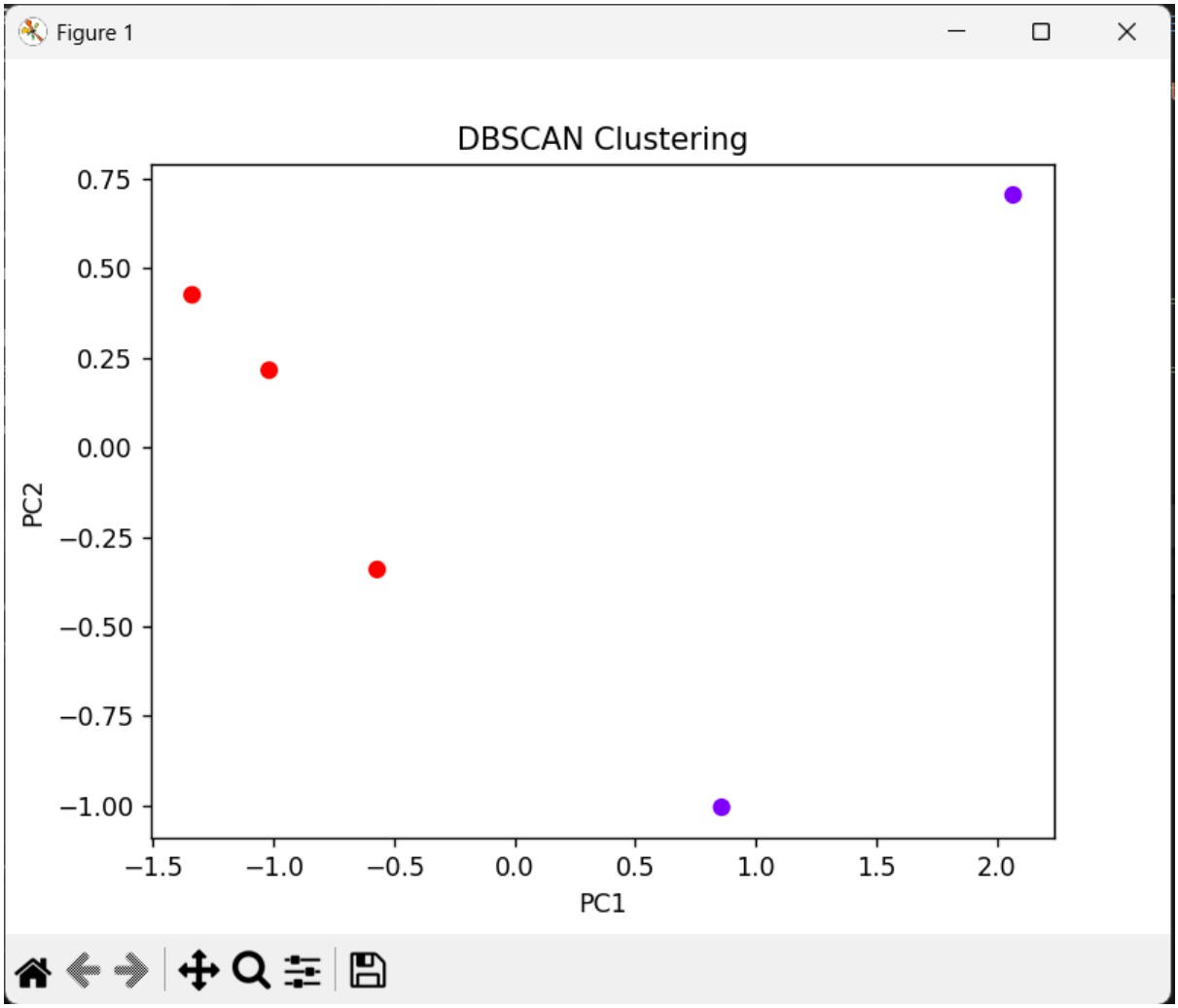
Figure 1
DBSCAN Clustering
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
PC2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
PC1