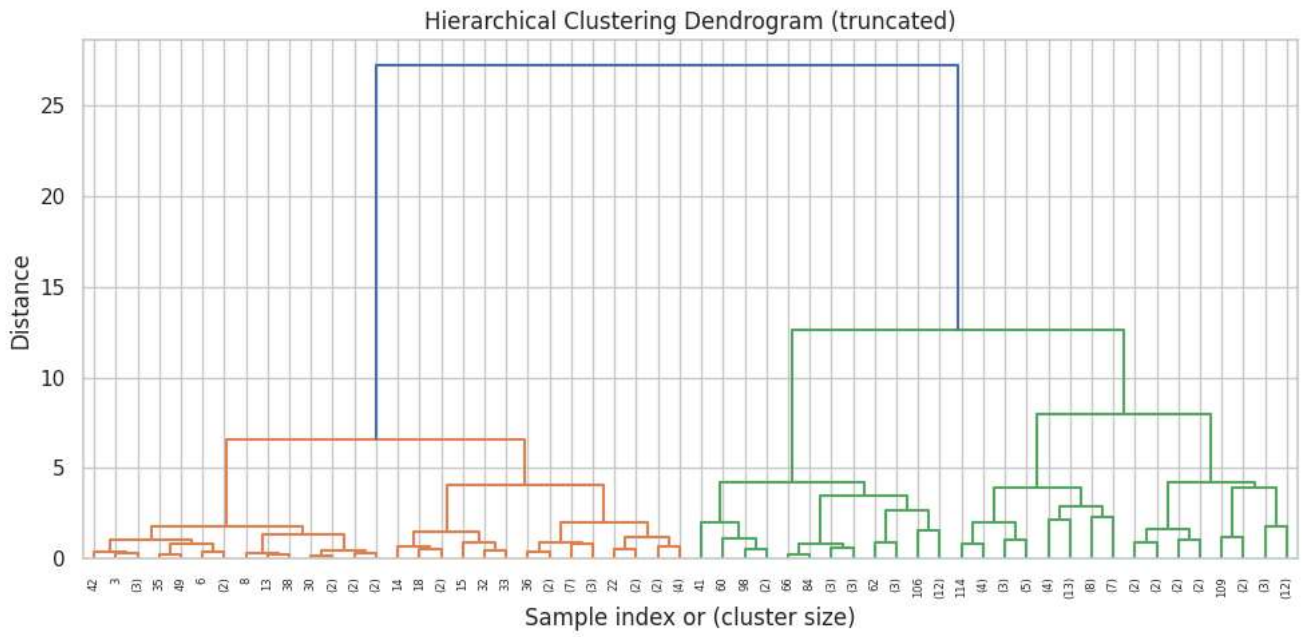
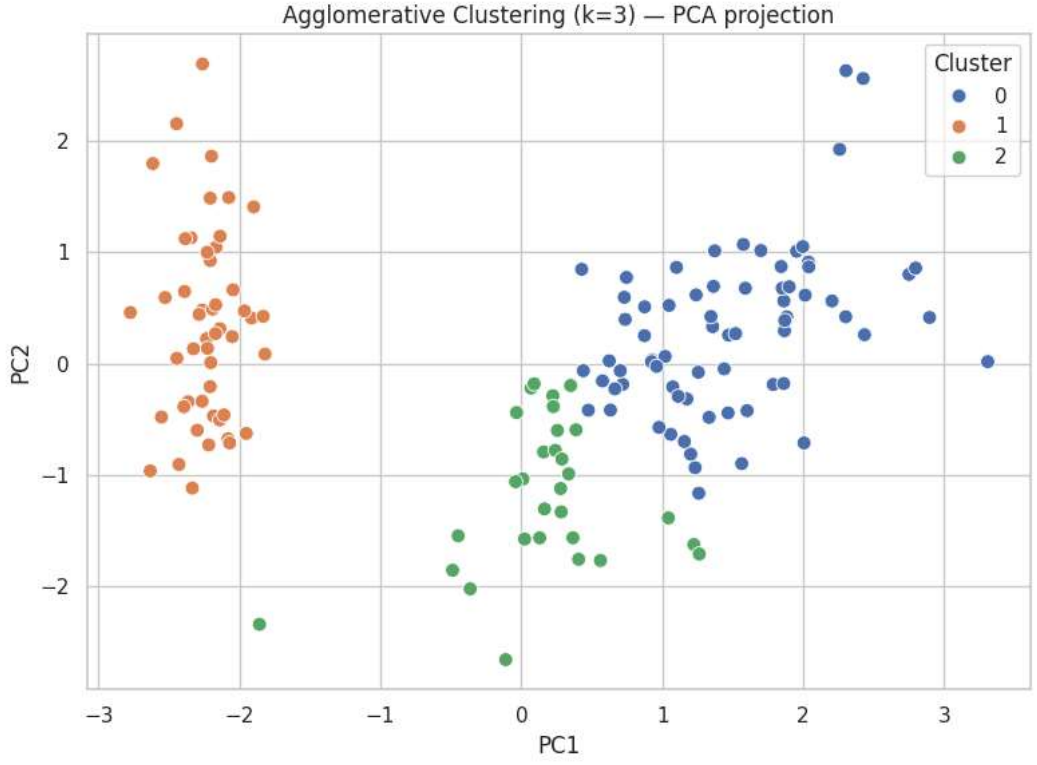
```
# hierarchical_clustering_example.py
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn import datasets
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import AgglomerativeClustering
from sklearn.metrics import silhouette_score

from scipy.cluster.hierarchy import linkage, dendrogram, fcluster

sns.set(style="whitegrid")

# 1) Load example data (Iris)
iris = datasets.load_iris()
X = iris.data
y_true = iris.target
feature_names = iris.feature_names

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

# 3) Optional: compute hierarchical linkage matrix for dendrogram
#    Common linkage methods: 'single', 'complete', 'average', 'ward'
Z = linkage(X_scaled, method='ward', metric='euclidean')

# 4) Plot dendrogram to inspect cluster structure
plt.figure(figsize=(10, 5))
dendrogram(Z, truncate_mode='level', p=5, show_leaf_counts=True)
plt.title('Hierarchical Clustering Dendrogram (truncated)')
plt.xlabel('Sample index or (cluster size)')
plt.ylabel('Distance')
plt.tight_layout()
plt.show()

# 5) Choose number of clusters (k). Example: k = 3
k = 3

# 6) Two ways to get clusters:
# A) Using scipy fcluster on linkage matrix:
labels_scipy = fcluster(Z, t=k, criterion='maxclust') - 1  # make 0-index

# B) Using sklearn AgglomerativeClustering:
agg = AgglomerativeClustering(n_clusters=k, linkage='ward')  # use same linkage
labels_sklearn = agg.fit_predict(X_scaled)

# They should be similar (labels may be permuted).
print("Unique clusters (scipy):", np.unique(labels_scipy))
print("Unique clusters (sklearn):", np.unique(labels_sklearn))

# 7) Evaluate clustering quality (silhouette score) - requires >= 2 clusters
print("Silhouette score (sklearn labels):", silhouette_score(X_scaled, labels_sklearn))

# 8) Visualize clusters in 2D using PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

plt.figure(figsize=(8, 6))
palette = sns.color_palette("deep", n_colors=k)
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=labels_sklearn,
                palette=palette, legend='full', s=60)
plt.title(f'Agglomerative Clustering (k={k}) — PCA projection')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.legend(title='Cluster')
plt.tight_layout()
plt.show()

# 9) Quick cluster summary (counts)
print(pd.Series(labels_sklearn).value_counts().sort_index())

# 10) If you want to cut the dendrogram at a distance threshold instead of k:
```

```
distance_threshold = 7.0  # example — pick by inspecting dendrogram vertical scale
labels_by_distance = fcluster(Z, t=distance_threshold, criterion='distance') - 1
print("Clusters from distance threshold:", np.unique(labels_by_distance))
```




```
Unique clusters (scipy): [0 1 2]
Unique clusters (sklearn): [0 1 2]
Silhouette score (sklearn labels): 0.4466890410285909
```




```
0    71
1    49
2    30
Name: count, dtype: int64
Clusters from distance threshold: [0 1 2 3]
```