**Problem Statement:**

Apply AGNES (single linkage, complete linkage, average linkage) on Iris dataset for clustering.

**Dataset Description:**

The Iris dataset is one of the most famous datasets in machine learning and statistics. It is widely used for classification and clustering tasks. The dataset consists of measurements of 150 iris flowers, where each sample corresponds to a single flower from one of three species.

Details of the Iris Dataset:

Number of samples: 150 (50 samples from each of 3 species)

Number of features: 4 features (measurements of the flowers)

Number of classes (species): 3 species

Species: Setosa, Versicolor, Virginica

Features (Columns):

Sepal Length (cm) – The length of the sepal.

Sepal Width (cm) – The width of the sepal.

Petal Length (cm) – The length of the petal.

Petal Width (cm) – The width of the petal.

**Procedure:**

Step 1: Initialize Each Data Point as a Separate Cluster.

Step 2: Compute the Distance Between Clusters using euclidean distance.

Step 3: Merge the Two Closest Clusters.

Step 4: Update the Distance Matrix according to the linkage criteria (Single, complete, average).

Step 5: Repeat steps 3 and 4 until all points are in a single cluster.

Step 6: The final result of AGNES is a dendrogram, which shows how clusters are merged over time.

**Source code:**

import numpy as np

import pandas as pd

from sklearn import datasets

from sklearn.preprocessing import StandardScaler, MinMaxScaler

from sklearn.cluster import AgglomerativeClustering

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

from scipy.cluster.hierarchy import linkage, dendrogram

# Load the Iris dataset

iris = datasets.load\_iris()

X = iris.data # Features (Sepal Length, Sepal Width, Petal Length, Petal Width)

y = iris.target # True labels (not used in unsupervised learning)

# Convert to DataFrame for better preprocessing

df = pd.DataFrame(X, columns=iris.feature\_names)

# Handle missing values (if any) by filling with mean values

df.fillna(df.mean(), inplace=True)

# Remove duplicate rows (if any)

df.drop\_duplicates(inplace=True)

# Standardize the features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(df)

# Define a function to apply AGNES with different linkage methods

def agnes\_clustering(linkage\_method):

agnes = AgglomerativeClustering(n\_clusters=3, linkage=linkage\_method)

cluster\_labels = agnes.fit\_predict(X\_scaled)

# Compute linkage matrix for dendrogram

Z = linkage(X\_scaled, method=linkage\_method)

# Reduce dimensionality for visualization using PCA

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X\_scaled)

# Plot the clusters

plt.figure(figsize=(8, 6))

for cluster in range(3):

plt.scatter(

X\_pca[np.where(cluster\_labels == cluster), 0],

X\_pca[np.where(cluster\_labels == cluster), 1],

label=f"Cluster {cluster + 1}"

)

plt.title(f"AGNES Clustering ({linkage\_method.capitalize()} Linkage)")

plt.xlabel("Principal Component 1")

plt.ylabel("Principal Component 2")

plt.legend()

plt.show()

# Plot the dendrogram

plt.figure(figsize=(10, 6))

dendrogram(Z)

plt.title(f"Dendrogram ({linkage\_method.capitalize()} Linkage)")

plt.xlabel("Data Points")

plt.ylabel("Distance")

plt.show()

# Apply AGNES with single linkage

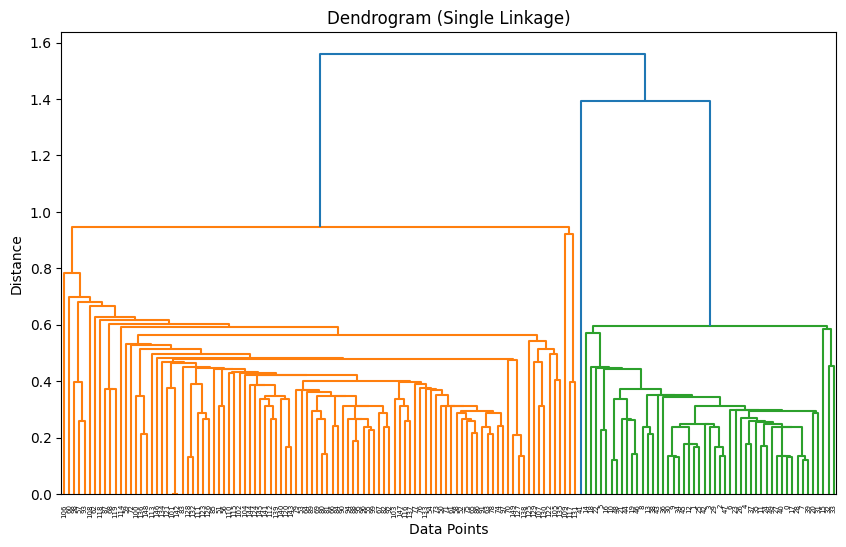
agnes\_clustering("single")

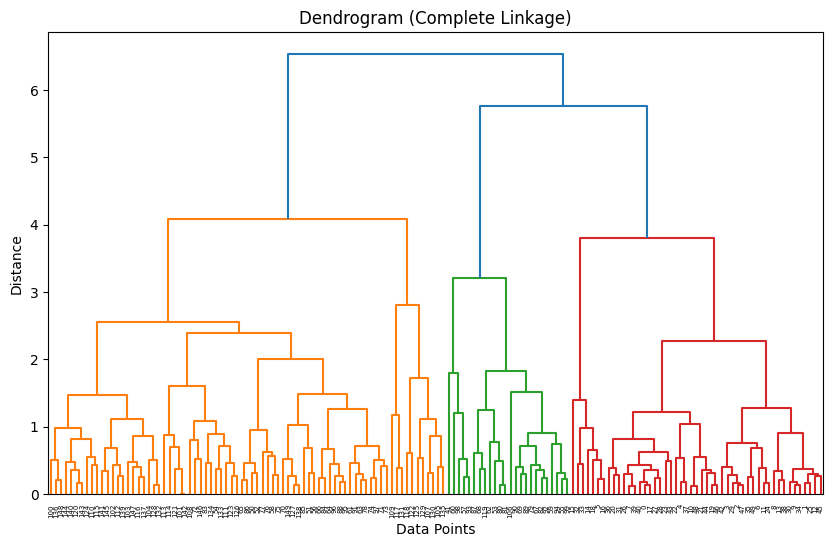
# Apply AGNES with complete linkage

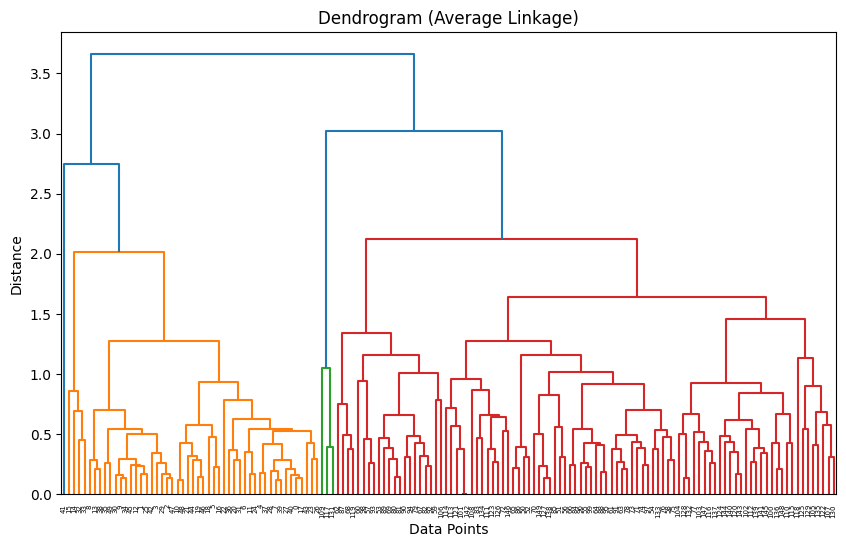
agnes\_clustering("complete")

# Apply AGNES with average linkage

agnes\_clustering("average")

**Output:**





**Discussion:**

Agglomerative Clustering (AGNES) is a hierarchical clustering algorithm. Here we have used PCA (Principal component analysis) component to reduce the dimension of the problem and make visualization easier. You can achieve same with Euclidian distance but you can not visualize the dataset.