```
# Step-1: Importing necessary libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import AgglomerativeClustering
import scipy.cluster.hierarchy as sch


# Step-2: Load the dataset

dataset = pd.read_csv("/content/CLUSTERING.csv")
dataset.describe()
```

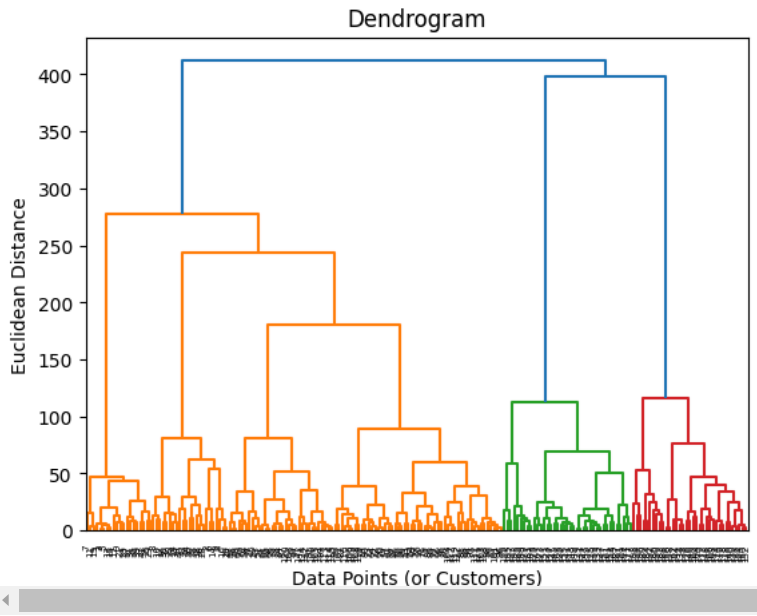
	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000
mean	38.850000	60.560000	50.200000
std	13.969007	26.264721	25.823522
min	18.000000	15.000000	1.000000
25%	28.750000	41.500000	34.750000
50%	36.000000	61.500000	50.000000
75%	49.000000	78.000000	73.000000
max	70.000000	137.000000	99.000000

```
# Step-3: Extracting the features
# Here we are assuming that all columns are features

x = dataset.iloc[:, :].values


# Step-4: Build the Dendrogram
# This helps visualize the hierarchical clustering process
# We start with each data point as a single cluster and merge them step by step.

dendrogram = sch.dendrogram(sch.linkage(x, method="ward"))
plt.title("Dendrogram")
plt.xlabel("Data Points (or Customers)")
plt.ylabel("Euclidean Distance")
plt.show()
```




```
# Step-5: Determine the number of clusters using the dendrogram (e.g., 5 clusters)
# Using Agglomerative Clustering to apply hierarchical clustering

HC = AgglomerativeClustering(n_clusters=5, linkage='ward')
predicted = HC.fit_predict(x)


# Step-6: Visualize the Clusters
# This plot shows clusters based on two selected features.
```

```
plt.figure(figsize=(10, 7))
plt.scatter(x[predicted == 0, 0], x[predicted == 0, 1], c='red', label='Cluster 1')
plt.scatter(x[predicted == 1, 0], x[predicted == 1, 1], c='blue', label='Cluster 2')
plt.scatter(x[predicted == 2, 0], x[predicted == 2, 1], c='green', label='Cluster 3')
plt.scatter(x[predicted == 3, 0], x[predicted == 3, 1], c='yellow', label='Cluster 4')
plt.scatter(x[predicted == 4, 0], x[predicted == 4, 1], c='orange', label='Cluster 5')
plt.title("Clusters of Customers")
plt.xlabel("Feature 1 (e.g., Annual Income)")
plt.ylabel("Feature 2 (e.g., Spending Score)")
plt.legend()
plt.show()
```