Computer Engineering Department National University of Technology Islamabad, Pakistan

Introduction to Data Mining

Practice Exercise 09

Name: Muhammad Sami Uddin Rafay

Roll Number: F18604013

Submitted To: Dr. Kamran Javed

Date: 31 January 2020

Practice Exercise 08

Unsupervised Machine Learning | Hierarchical Clustering

Objective:

- Implement Hierarchical Clustering Algorithm in Python.

Equipment/Software Required:

- Python (Spyder 4.0 Anaconda Distribution)

Tasks:

1. Explain what is Hierarchical clustering and how we choose the number of clusters?
2. Write main steps of Hierarchical clustering
3. Reproduce the given code of Hierarchical clustering (using clust.xlsx file) and explain/comment each line of code i.e., input, output attributes and function.
4. Plot 2D and 3D views of given data and label both figures.
5. Implement Hierarchical for fisher iris data (Load fisher iris dataset with petal lengths and petal widths)
6. Determine the number of clusters for fisher data.
7. Plot clusters and cluster centers according to dendrogram,

Answers:

1. **Explain what is Hierarchical clustering and how we choose the number of clusters?**

   Hierarchical methods form the backbone of cluster analysis. As the name suggests, Hierarchical clustering is an algorithm that builds hierarchy of clusters. This algorithm starts with all the data points assigned to a cluster of their own. Then two nearest clusters are merged into the same cluster. In the end, this algorithm terminates when there is only a single cluster left.

   The need for hierarchical clustering naturally emerges in domains where it is not only required to discover similarity-based groups but also need to organize them.

   To determine clusters, we make horizontal cuts across the branches of the dendrogram. The number of clusters is then calculated by the number of vertical lines on the dendrogram, which lies under horizontal line.

   To decide where to cut a dendrogram? Practically, analysts do it based on their judgement and business need.

2. **Write main steps of Hierarchical clustering.**
   Main Steps of Hierarchical clustering:
   i. Compute the proximity matrix, if necessary.
   ii. repeat
   iii. Merge the closest two clusters.
   iv. Update the proximity matrix to reflect the proximity between the new cluster and the original clusters.

V. until Only one cluster remains.

The rest of answers are implemented below: -

Code (clust.xlsx):

 # importing Libraries

```
import numpy as np
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram, linkage
import pandas as pd
```

 # Load the clust dataset

```
clust=pd.read_excel(r"C:\Users\User\Desktop\clust.xlsx")
```

 # specifying column names for clust

```
attributes = ["X", "Y", "Z"]
clust.columns = attributes
X=round(clust["X"],1)
Y=round(clust["Y"],1)
Z=clust["Z"]

C=np.array(list(zip(X,Y,Z)))
Z = linkage(C[:,:2], 'ward')

labels = C
plt.figure(1,figsize=(10, 7))
plt.subplots_adjust(bottom=0.1)
plt.scatter(C[:,0],C[:,1], label='True Position')
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
for label, x, y in zip(labels, C[:, 0], C[:, 1]):
    plt.annotate(
        label,
        xy=(x, y), xytext=(-3, 3),
        textcoords='offset points', ha='right', va='bottom', fontsize=8)
plt.show()
```

 # Using AgglomerativeClustering command from sklearn

```
cluster = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
cluster.fit_predict(C)
print(cluster.labels_)
```

 # 2D plot of clust dataset

```
plt.figure(2,figsize=(10, 7))
plt.scatter(C[:,0],C[:,1], c=cluster.labels_, cmap='rainbow')
plt.xlabel("X")
```

```
plt.ylabel("Y")
plt.show()
```

 # 3D plot of clust dataset

```
fig2 = plt.figure(3)
bx = Axes3D(fig2)
bx.scatter(C[:, 0], C[:, 1], C[:,2],c=cluster.labels_,cmap='rainbow')
bx.set_xlabel("petal Length")
bx.set_ylabel("petal width")
bx.set_zlabel("sepal width")
plt.show()
```

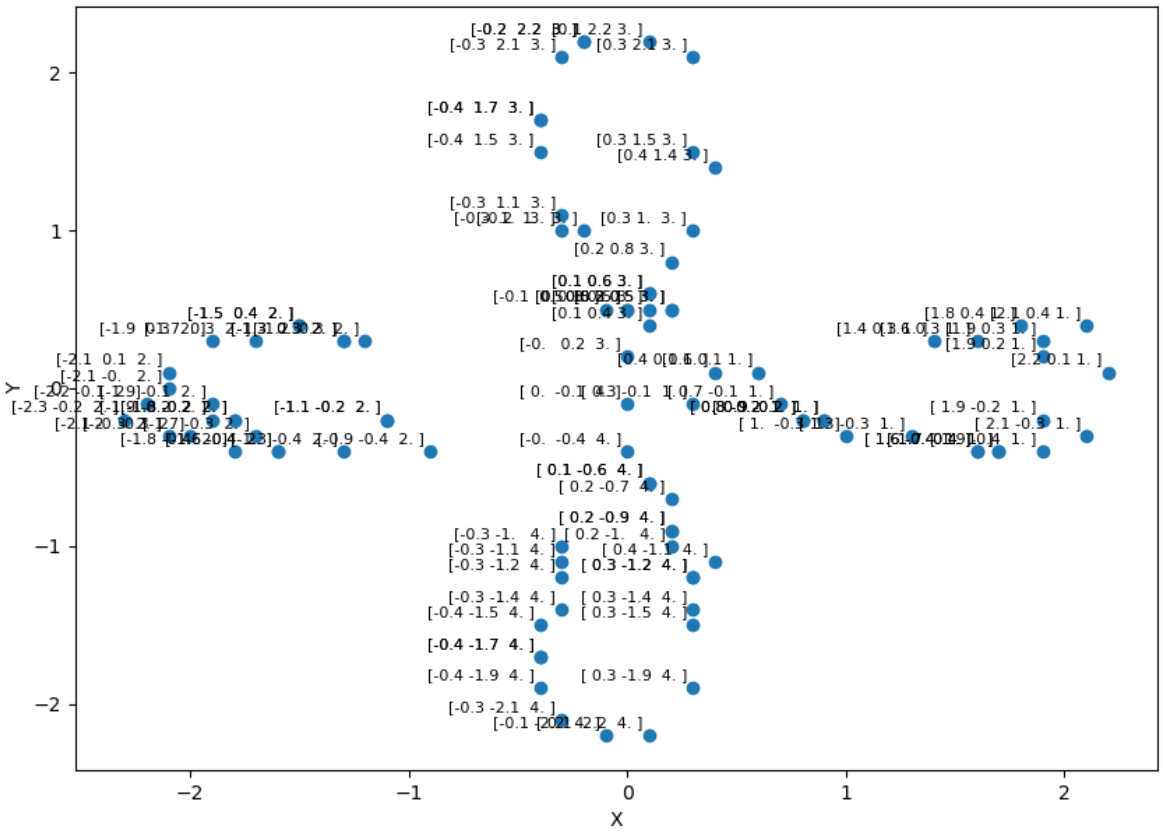
 # plotting clusters according to Dendrogram

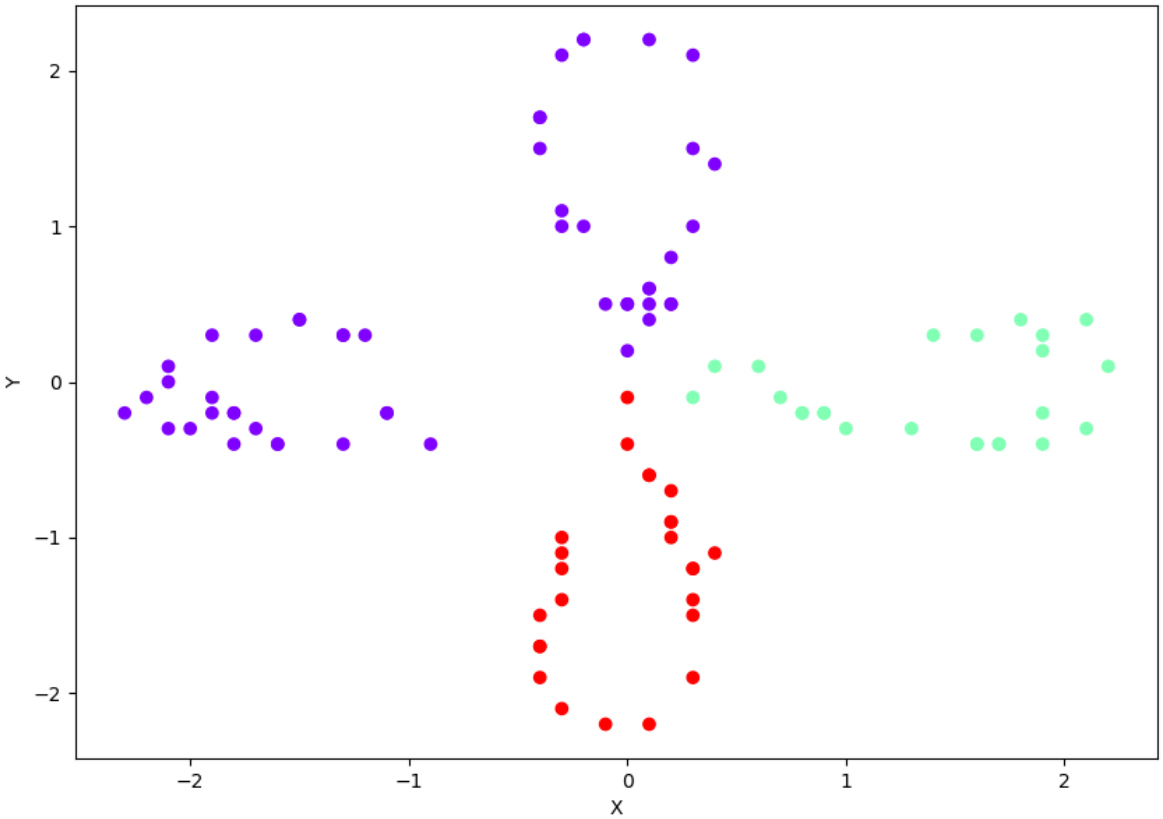
```
plt.figure(4,figsize=(10, 7))
dendrogram(Z,
        orientation='top',
        labels=cluster.labels_,
        distance_sort='descending',
        show_leaf_counts=True)
plt.title("Hierarchical Clustering Dendogram Representation of clust Dataset")
plt.show()
```

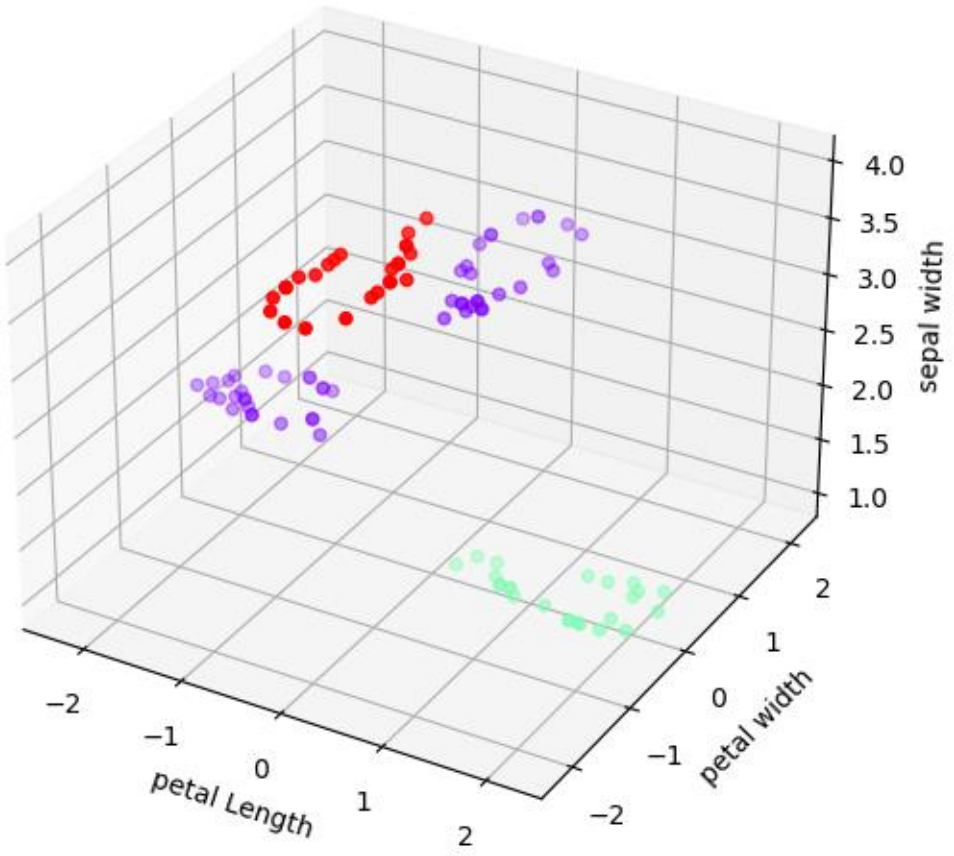
Output:

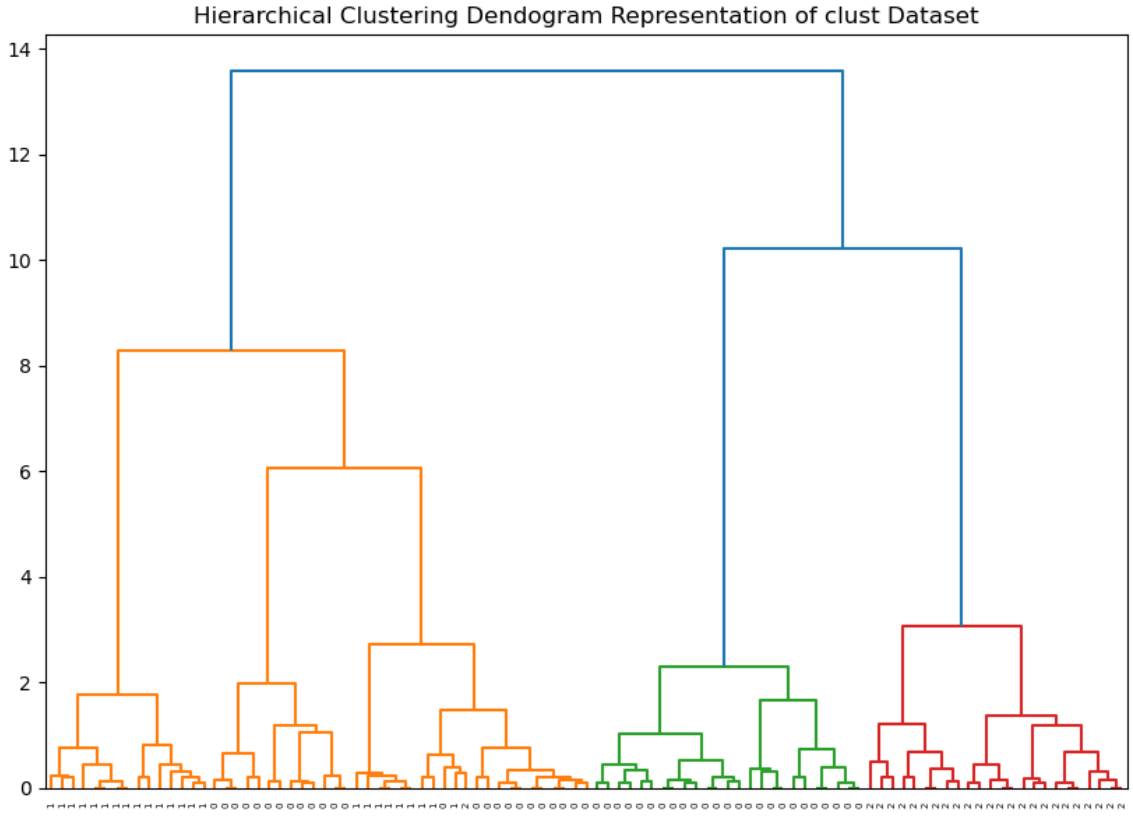
```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

Graphs:






4.0
3.5
3.0
2.5
2.0
1.5
1.0
sepal width
−2
−1
0
1
2
petal Length
2
1
0
−1
−2
petal width


Hierarchical Clustering Dendogram Representation of clust Dataset
14
12
10
8
6
4
2
0

Code (fisher iris):

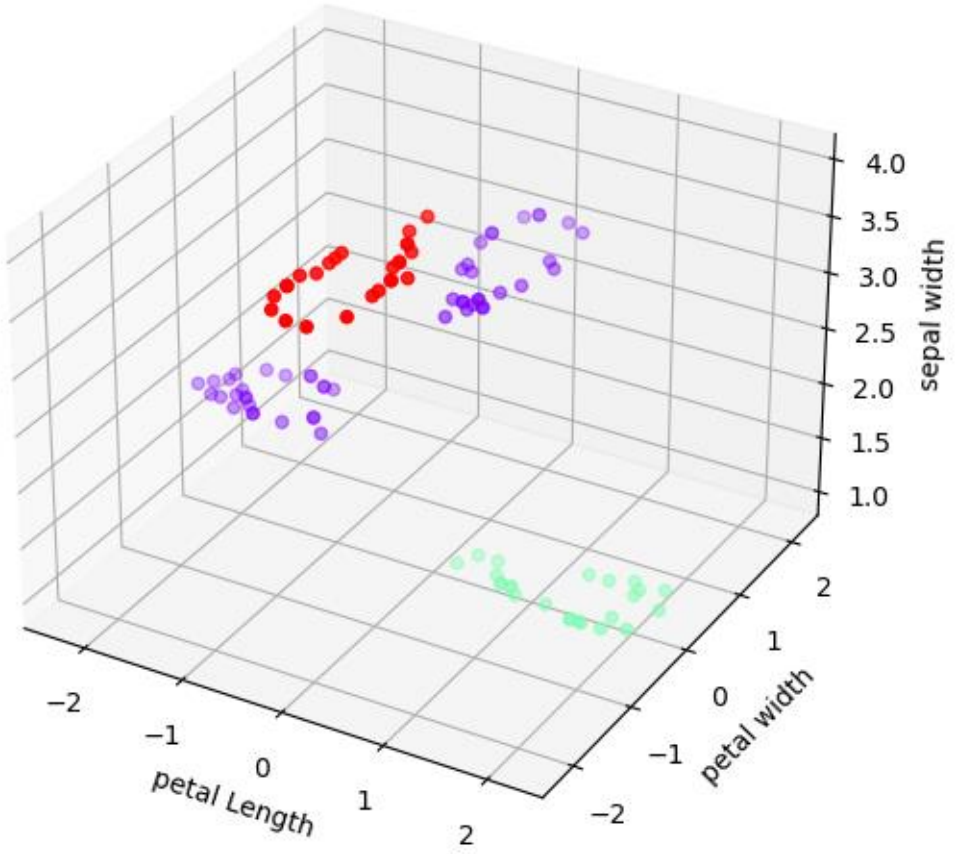
```
# importing libraries

import numpy as np
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram, linkage
import pandas as pd

# load fisher iris dataset

fisher_iris=pd.read_csv(r"C:\Users\User\Downloads\iris.data")
attributes = ["sepal_length", "sepal_width", "petal_length", "petal_width", "class"]
fisher_iris.columns = attributes

# specifying column names for fisher iris

petal_length=fisher_iris["petal_length"]
petal_width=fisher_iris["petal_width"]
sepal_length=fisher_iris["sepal_length"]
sepal_width=fisher_iris["sepal_width"]
X=np.array(list(zip(petal_length,petal_width,sepal_width)))

# Using Linkage command from sklearn

Z = linkage(X[:,:2], 'ward')

# 2D plotting of iris dataset

labels = X
plt.figure(1,figsize=(10, 7))
plt.subplots_adjust(bottom=0.1)
plt.scatter(X[:,0],X[:,1], label='True Position')
plt.xlabel("petal Length")
plt.ylabel("petal Width")
plt.show()

# annotating/labelling the datapoint on graphs according to x-y values

for label, x, y in zip(labels, X[:, 0], X[:, 1]):
    plt.annotate(
        label,
        xy=(x, y), xytext=(-3, 3),
        textcoords='offset points', ha='right', va='bottom', fontsize=8)
plt.show()

# using AgglomerativeClustering command from sklearn to calculate clusters
cluster = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
cluster.fit_predict(X)
print(cluster.labels_)
plt.figure(2,figsize=(10, 7))
plt.scatter(X[:,0],X[:,1], c=cluster.labels_, cmap='rainbow')
plt.xlabel("petal Length")
```

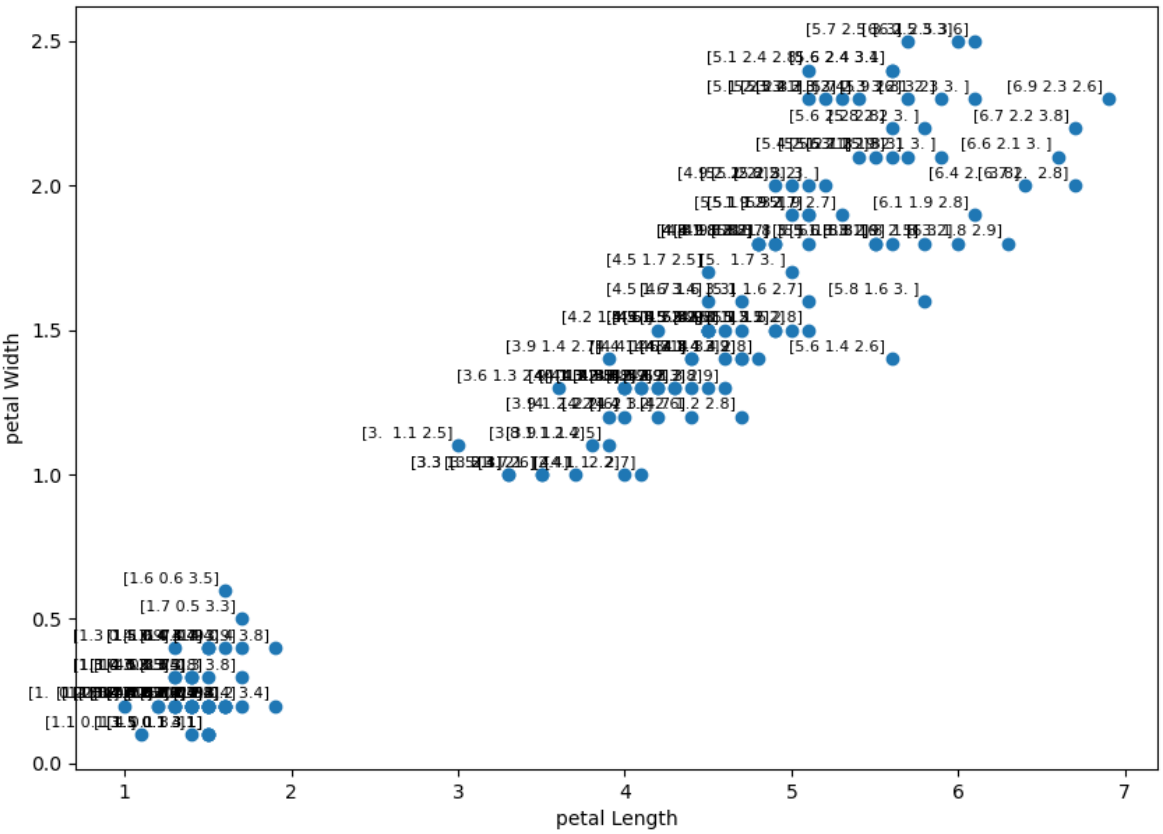
```
plt.ylabel("petal Width")
plt.show()

# 3D plot of iris dataset

fig2 = plt.figure(4)
bx = Axes3D(fig2)
bx.scatter(X[:, 0], X[:, 1], X[:,2],c=cluster.labels_,cmap='rainbow')
bx.set_xlabel("petal Length")
bx.set_ylabel("petal width")
bx.set_zlabel("sepal width")
plt.show()

# plotting dendrogram of iris clusters

plt.figure(5,figsize=(10, 7))
dendrogram(Z,
        orientation='top',
        labels=cluster.labels_,
        distance_sort='descending',
        show_leaf_counts=True)
plt.title("Hierarchical Clustering Dendogram Representation of Iris Dataset")
plt.show()
```

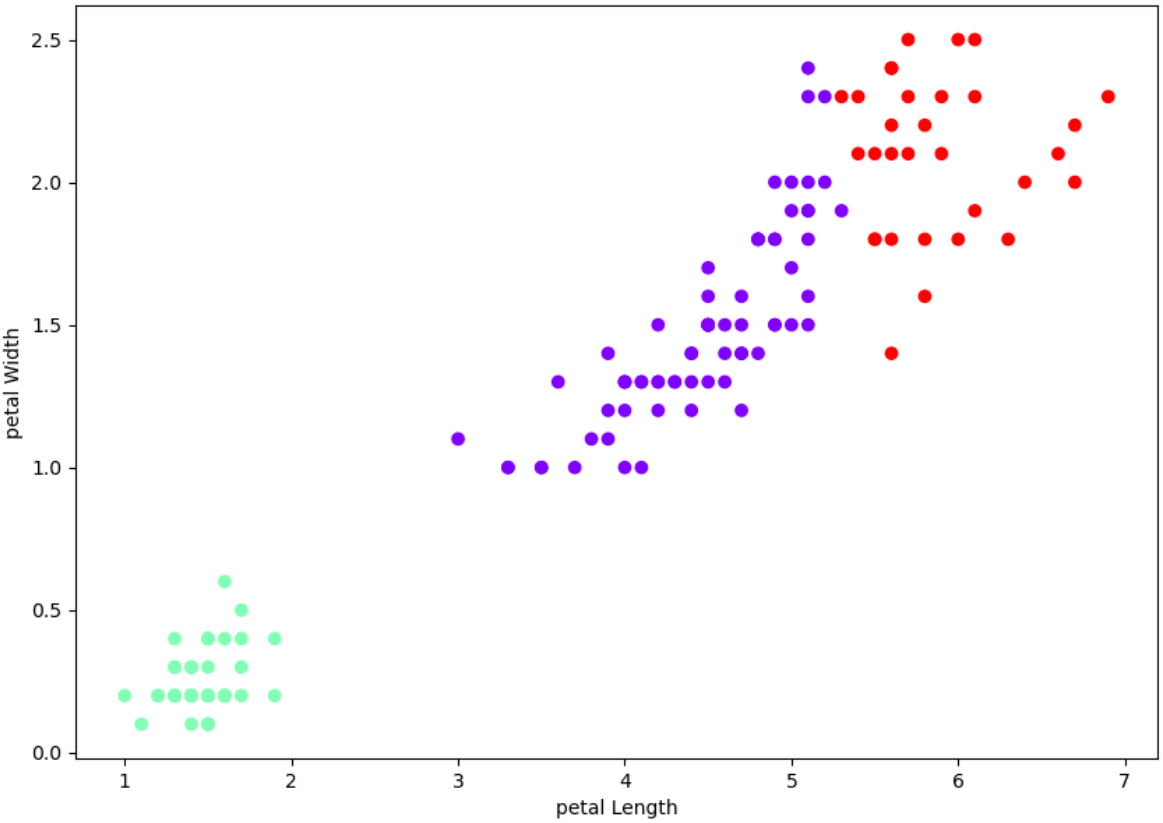
Output:

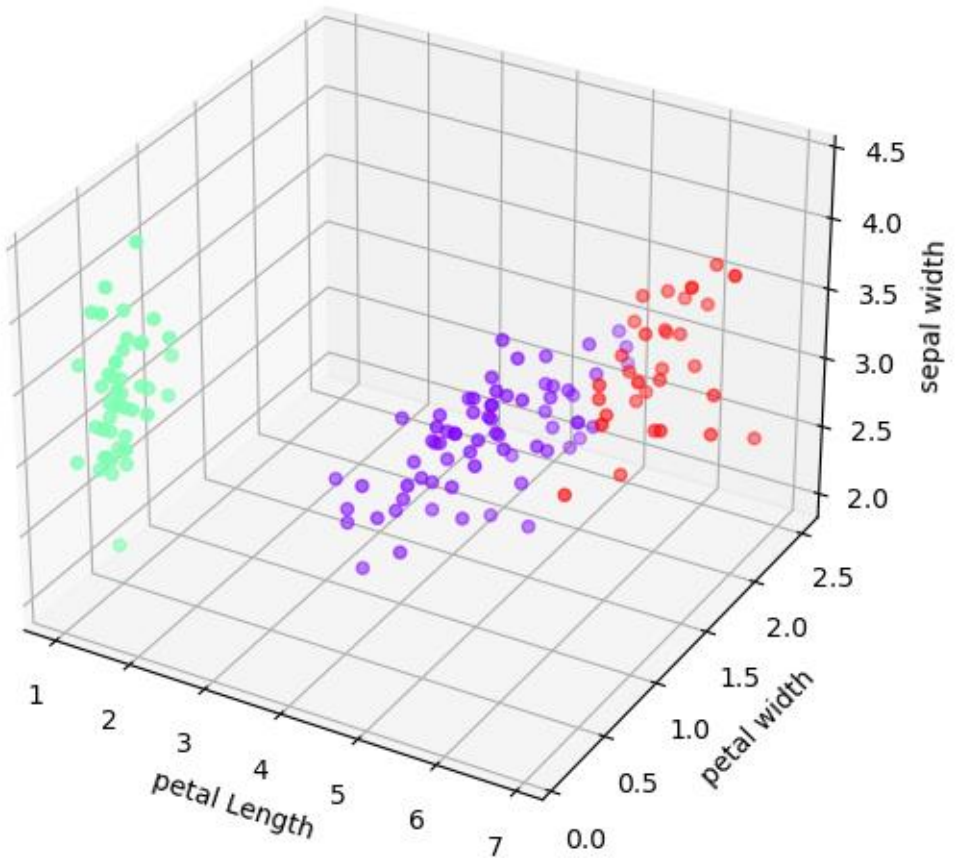
```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 2 2 2 0 2 2 2 0 0
 2 0 0 2 2 2 2 0 2 0 2 0 2 2 0 0 2 2 2 2 2 0 2 2 2 2 0 2 2 0 0 2 2 0 0 0 2
 0]
```

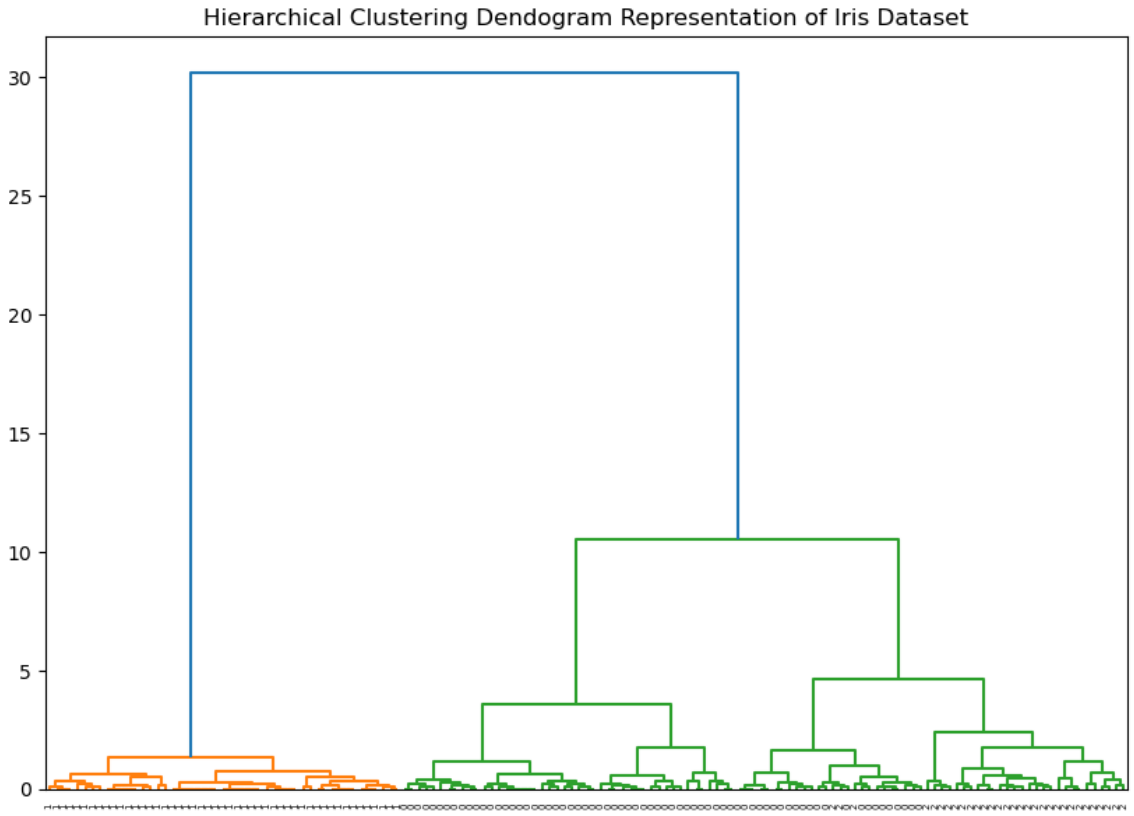
Graphs:


4.0
3.5
3.0
2.5
2.0
1.5
1.0
sepal width
−2
−1
0
1
2
petal Length
2
1
0
−1
−2
petal width


petal Width
petal Length
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7


petal Width
petal Length
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7


sepal width
petal width
petal Length
4.5
4.0
3.5
3.0
2.5
2.0
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7



Results and Discussions:

Hierarchical clustering is an algorithm that builds hierarchy of clusters. This algorithm starts with all the data points assigned to a cluster of their own. Then two nearest clusters are merged into the same cluster. In the end, this algorithm terminates when there is only a single cluster left.

Conclusion:

The need for hierarchical clustering naturally emerges in domains where it is not only required to discover similarity-based groups but also need to organize them.