Academic Year: 2025-26 **Semester: V**
Class/Branch: T.E. DS **Subject: DWMLab**

EXPERIMENT NO. 5

1. **Aim:** Implementation of Agglomerative hierarchical clustering algorithm using python.

2. **Objectives:** From this experiment, the student will be able to

- Learn about the Hierarchical Clustering technique.
- Learn to plot the results in a dendrogram.

3. **Theory:**

Clustering:

The technique of segregating given datasets into various groups based on similar features and characteristics is called clustering. The groups being formed are known as clusters.

Hierarchical Clustering:
In data mining, hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. Strategies for hierarchical clustering generally fall into two types:

Agglomerative: This is a "bottom up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

Divisive: This is a "top down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

The implementation and working of the agglomerative (bottom-up) clustering algorithm are explained in the steps below:

1. Start by considering each data point as its own singleton cluster.
2. After each iteration of calculating Euclidian distance, merge two clusters with minimum distance.
3. Stop when there is a single cluster of all examples, else go to step 2

Code:

1. Import the modules.

Take input and start by visualizing some data points:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering
```

2. Create arrays that resemble two variables in a dataset.

```
x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]
```

3. Turn the data into a set of points:

```
data = list(zip(x, y))
print(data)
```

```
[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24),
(6, 22), (10, 21), (12, 21)]
```

4. Compute the linkage between all of the different points. Here we use a simple euclidean distance measure and Ward's linkage, which seeks to minimize the variance between clusters.

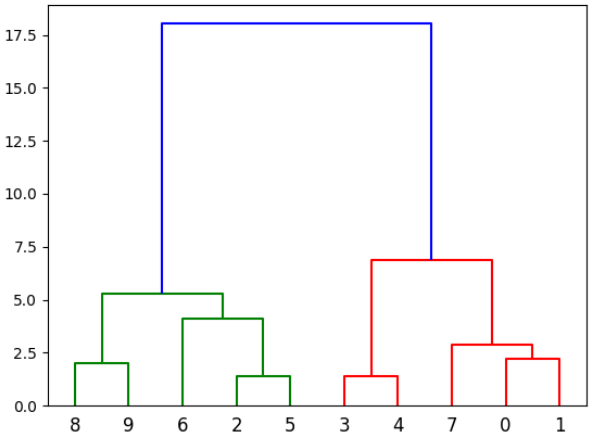
```
linkage_data = linkage(data, method='single', metric='euclidean')
```

5. Plot the results in a dendrogram.

This plot will show us the hierarchy of clusters from the bottom (individual points) to the top (a single cluster consisting of all data points).

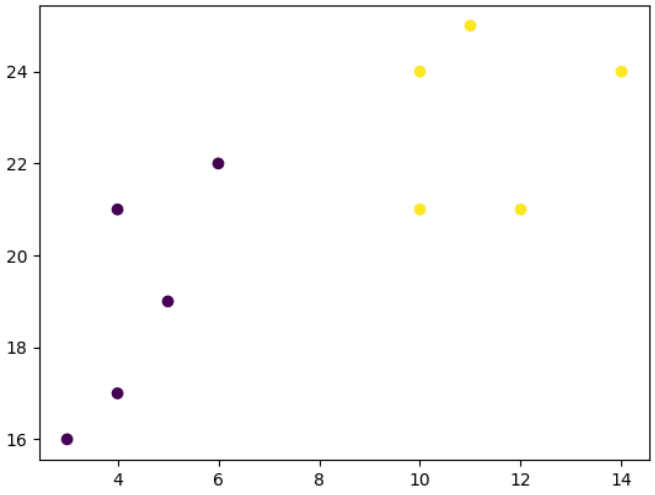
plt.show() lets us visualize the dendrogram instead of just the raw linkage data.

```
dendrogram(linkage_data)
plt.show()
```



6. Finally, if we plot the same data and color the points using the labels assigned to each index by the hierarchical clustering method, we can see the cluster each point was assigned to:

```
plt.scatter(x, y, c=labels)
plt.show()
```



4. Conclusion: