Lecture 10

Hierarchical Clustering
Agglomerative Clustering
Spectral Clustering

Hierarchical clustering is a popular clustering technique used in data mining and machine learning. It organizes data points into a hierarchical structure of clusters, often represented as a tree-like diagram called a dendrogram. Hierarchical clustering can be categorized into two main approaches: agglomerative (bottom-up) and divisive (top-down). Here, we'll focus on the agglomerative approach, which is more commonly used.

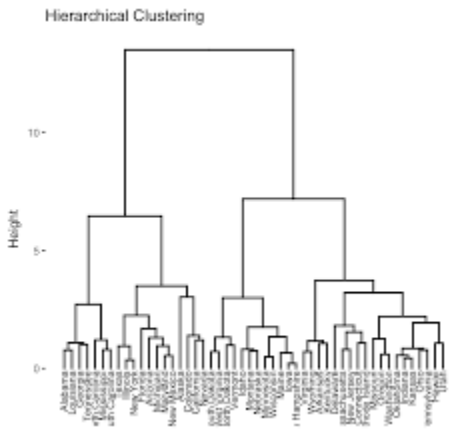
Agglomerative Hierarchical Clustering: Agglomerative hierarchical clustering starts with each data point as its cluster and then recursively merges the closest clusters, forming a hierarchy. Here's a step-by-step overview of how it works:

1. Initialization: Begin with each data point as a separate cluster. If you have n data points, you start with n clusters, each containing a single data point.

2. Merge Closest Clusters: Identify the two closest clusters based on a distance metric, often using methods like single-linkage, complete-linkage, or average-linkage. Merge these two clusters into a new, larger cluster.

3. Update Distance Matrix: Recalculate the distances between this newly formed cluster and all other clusters, resulting in an updated distance matrix.

4. Repeat: Repeat steps 2 and 3 until there is only one cluster left, which contains all data points. The hierarchy of clusters forms a dendrogram. *Dendrogram*: A dendrogram is a tree-like diagram that represents the hierarchical structure of clusters. The vertical lines in the dendrogram indicate the merging of clusters at each step. The height at which clusters merge reflects the dissimilarity between them.



Key Considerations:

Choice of Distance Metric: The choice of distance metric (Euclidean, Manhattan, etc.) significantly affects the results. The method for measuring the distance between clusters also matters, as single-linkage, complete-linkage, and average-linkage can lead to different outcomes.

Number of Clusters: Hierarchical clustering does not require you to specify the number of clusters (k) beforehand. Instead, you can choose the desired level of granularity by cutting the dendrogram at a certain height.

Interpreting the Dendrogram: The dendrogram provides insights into the hierarchical relationships between clusters. By cutting it at various heights, you can obtain different clusterings with varying numbers of clusters.

Scalability: Hierarchical clustering can be computationally intensive, especially for large datasets, as it involves repeated distance calculations and matrix updates.

Applications:

- Hierarchical clustering is widely used in various fields and applications, including:
- Biology: For phylogenetic tree construction and gene expression analysis.
- Image Analysis: For segmentation and object detection.
- Marketing: For customer segmentation and market basket analysis.
- Social Network Analysis: For community detection.
- Document Clustering: For organizing documents into topic hierarchies.

Hierarchical clustering is a versatile technique that provides insights into the structure of data, making it valuable for exploratory data analysis and pattern discovery. It allows you to visualize the relationships between clusters and analyze data at multiple levels of granularity.

An example of hierarchical clustering in R using the hclust function and the iris dataset. In this example, we will perform agglomerative hierarchical clustering on the sepal length and sepal width measurements from the iris dataset. Hierarchical clustering will create a dendrogram, allowing us to explore the hierarchical relationships between data points.

Here's how to perform hierarchical clustering in R:

```
# Load the iris dataset
data(iris)
# Select the relevant features (sepal length and sepal width)
iris_features <- iris[, c("Sepal.Length", "Sepal.Width")]
# Perform hierarchical clustering
hc <- hclust(dist(iris_features), method = "complete")
# Complete-linkage clustering # Plot the dendrogram
plot(hc, main = "Hierarchical Clustering of Iris Data", xlab =
"Species")
```

This code does the following:
Loads the iris dataset and selects the relevant features (sepal length and sepal width). Performs hierarchical clustering using the hclust function. We use the "complete" linkage method, which calculates the distance between clusters as the maximum distance between their individual data points.

You can choose other linkage methods like "single," "average," or "ward.D2" based on your preferences. It then plots the dendrogram using the plot function, which shows the hierarchical structure of the clusters. In the dendrogram, data points are represented as leaves, and the height at which branches merge indicates the dissimilarity between clusters.

When you run this code, you'll see a dendrogram that illustrates the hierarchical clustering of the iris data. Data points are grouped together based on their sepal length and sepal width measurements, and you can choose to cut the dendrogram at a certain height to obtain different clusterings with varying numbers of clusters. This hierarchical approach provides insights into the relationships between clusters and allows you to explore the data at multiple levels of granularity.

Agglomerative clustering is a hierarchical clustering technique that falls under the broader category of hierarchical clustering algorithms. Unlike divisive clustering, which starts with all data points in one cluster and recursively divides them into smaller clusters, agglomerative clustering begins with each data point in a separate cluster and progressively merges them into larger clusters. Here's how agglomerative clustering works:

Basic Procedure:
Initialization: Start with each data point as a single cluster. If you have n data points, you begin with n clusters, each containing one data point.

Merge Closest Clusters: Identify the two closest clusters based on a chosen linkage method, which determines how to measure the distance between clusters. Common linkage methods include:

- *Single Linkage*: Based on the minimum distance between data points in the two clusters.
- *Complete Linkage*: Based on the maximum distance between data points in the two clusters.
- *Average Linkage*: Based on the average distance between data points in the two clusters.
- *Ward's Linkage*: Minimizes the increase in variance within clusters when merging.

Update Distance Matrix: Recalculate the distances between the newly formed cluster and all other clusters, resulting in an updated distance matrix.

Repeat: Continue merging the two closest clusters and updating the distance matrix until there is only one cluster left, which contains all data points. The hierarchy of clusters forms a dendrogram.

Dendrogram: The output of agglomerative clustering is often visualized as a dendrogram, which is a tree-like diagram that represents the hierarchy of clusters. The vertical lines in the dendrogram indicate the merging of clusters at each step, with the height at which they merge reflecting the dissimilarity between them.

Key Considerations:
Agglomerative clustering does not require you to specify the number of clusters (k) in advance. Instead, you can choose the desired level of granularity by cutting the dendrogram at a certain height.

The choice of linkage method can significantly affect the results. Different linkage methods may lead to different cluster shapes and sizes.

The order in which clusters are merged can impact the final clustering result. The algorithm can start with the two closest clusters or the two farthest clusters.
Agglomerative clustering is computationally intensive, especially for large datasets, as it involves repeated distance calculations and matrix updates.

Applications:
Agglomerative clustering is used in various fields and applications, including biology for phylogenetic tree construction, image analysis for segmentation and object detection, marketing for customer segmentation, social network analysis for community detection, and document clustering for organizing documents into topic hierarchies.

Agglomerative clustering is a versatile technique that provides insights into the structure of data and allows for hierarchical exploration of clusters. It is particularly valuable when you want to understand relationships between clusters at different levels of granularity.

An example of agglomerative clustering in R using the agnes function from the cluster package. In this example, we will use a synthetic dataset and apply agglomerative clustering to group the data points into clusters. We'll also visualize the results using a dendrogram.

First, ensure that you have the cluster package installed. You can install it using the following command if it's not already installed:

```
install.packages("cluster")
```

Now, let's create a synthetic dataset and perform agglomerative clustering:

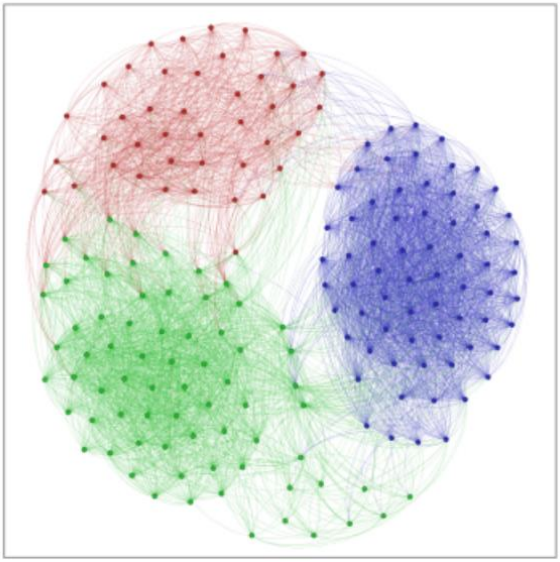
```
# Load the cluster library
library(cluster)
# Generate synthetic data
set.seed(123)
data <- matrix(rnorm(200, mean = 0, sd = 1), ncol = 2)
# Perform agglomerative clustering
agglo_result <- agnes(data, method = "complete")
# Using complete linkage
# Plot the dendrogram
plot(agglo_result, main = "Agglomerative Clustering Example")
```

In this code, we: Load the cluster library. Generate a synthetic dataset with two clusters using random data. Perform agglomerative clustering using the agnes function. We use the "complete" linkage method, which calculates the distance between clusters as the maximum distance between their individual data points. You can choose other linkage methods such as "single" or "average" based on your preferences. Plot the dendrogram using the plot function to visualize the hierarchical clustering results.

When you run this code, you will see a dendrogram that illustrates the hierarchical clustering of the synthetic data. The dendrogram shows the hierarchical structure of clusters, with the height at which branches merge indicating the dissimilarity between clusters.

You can choose to cut the dendrogram at a certain height to obtain different clusterings with varying numbers of clusters. Agglomerative clustering provides a hierarchical view of the data's clustering structure, allowing you to explore the data at different levels of granularity.

Spectral clustering is a powerful and versatile clustering technique that is widely used in data mining and machine learning. It differs from traditional clustering methods like K-Means or hierarchical clustering by embedding the data into a lower-dimensional space and performing clustering in that space. Spectral clustering is particularly useful for discovering clusters with complex shapes, handling non-convex clusters, and finding clusters of varying sizes. Here's how spectral clustering works:



Basic Procedure:
Affinity Matrix: Start with your data, which can be represented as a matrix of pairwise affinities or similarities between data points. The affinity matrix encodes how similar or related each data point is to every other data point. Common similarity measures include Euclidean distance, cosine similarity, or other kernel functions.

Graph Construction: Use the affinity matrix to construct a graph where data points are nodes, and the affinities represent edge weights. You can choose to build a fully connected graph or a k-nearest neighbor graph, depending on the nature of your data.

Spectral Embedding: Compute the Laplacian matrix of the graph, which characterizes the connectivity and structure of the data. This matrix can be used to derive the graph Laplacian eigenvectors.

Eigenvector Decomposition: Compute the eigenvectors and eigenvalues of the Laplacian matrix. These eigenvectors represent the embedded representation of the data in a lower-dimensional space.

Cluster Assignment: After obtaining the eigenvectors, perform clustering on the embedded data using conventional clustering techniques like K-Means or normalized cuts. You can choose the number of clusters (k) or use techniques like the eigengap heuristic to determine the optimal k.

Key Considerations:

- Spectral clustering can handle complex cluster shapes and is less sensitive to the initialization of cluster centers compared to K-Means.

- It can uncover clusters with varying densities and sizes.
- The choice of the similarity metric and the graph construction method is crucial, and it can affect the results significantly. Different choices can lead to different cluster structures.
- Spectral clustering may require parameter tuning, such as the number of neighbors in the k-nearest neighbor graph or the number of eigenvectors to use.
- It is suitable for both numerical and categorical data.
- Spectral clustering is computationally intensive, especially for large datasets, due to the eigenvector decomposition step.

Applications: Spectral clustering is used in various applications, including:

- *Image segmentation*: For partitioning images into regions with similar properties.
- *Document clustering*: For organizing documents into meaningful groups based on their content.
- *Biological data analysis*: For clustering genes or proteins in genomic and proteomic studies.
- *Community detection in social networks*: For identifying communities or groups of nodes with similar properties.
- *Anomaly detection*: For identifying data points that deviate from the norm in a dataset.

Resources:

1. https://www.r-bloggers.com/2016/01/hierarchical-clustering-in-r-2/
2. https://uc-r.github.io/hc_clustering
3. https://www.datacamp.com/tutorial/hierarchical-clustering-R
4. https://www.geeksforgeeks.org/hierarchical-clustering-in-r-programming/
5. https://www.statology.org/hierarchical-clustering-in-r/
6. https://bookdown.org/rdpeng/exdata/hierarchical-clustering.html
7. https://medium.com/@sukmaanindita/wards-hierarchical-clustering-method-using-r-studio-a47b5a79cb4d
8. https://online.stat.psu.edu/stat508/lesson/12/12.8
9. https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/
10. https://mhahsler.github.io/Introduction_to_Data_Mining_R_Examples/book/clustering-analysis.html
11. https://www.di.fc.ul.pt/~jpn/r/spectralclustering/spectralclustering.html
12. https://rpubs.com/nurakawa/spectral-clustering
13. https://rpubs.com/gargeejagtap/SpectralClustering