BIRCH Clustering

- **BIRCH Clustering:**
- BIRCH operates by first clustering the data points into subclusters, known as Clustering Features (CFs), and then clustering these CFs hierarchically. It's particularly useful for large datasets due to its ability to efficiently handle and summarize large amounts of data.

- **How BIRCH Works:**

1. **CF Construction**:

   BIRCH builds a tree-like structure where each non-leaf node represents a CF, which contains summarized information about the data points it covers, such as the centroid and the number of points.

2. **Clustering Feature (CF)**:

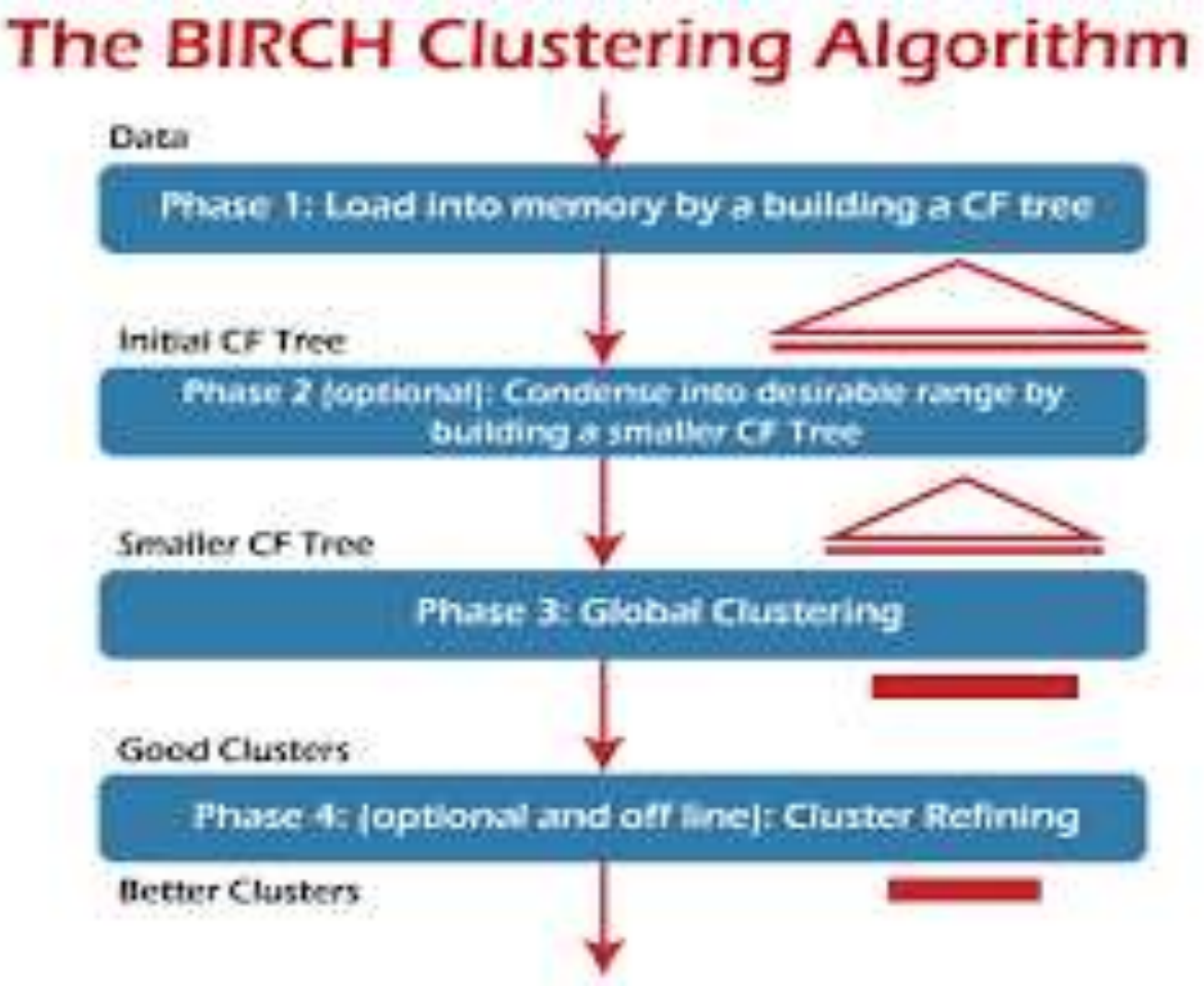
   CFs contain information about data points within a certain proximity, allowing BIRCH to summarize the dataset in a memory-efficient manner.

BIRCH Clustering

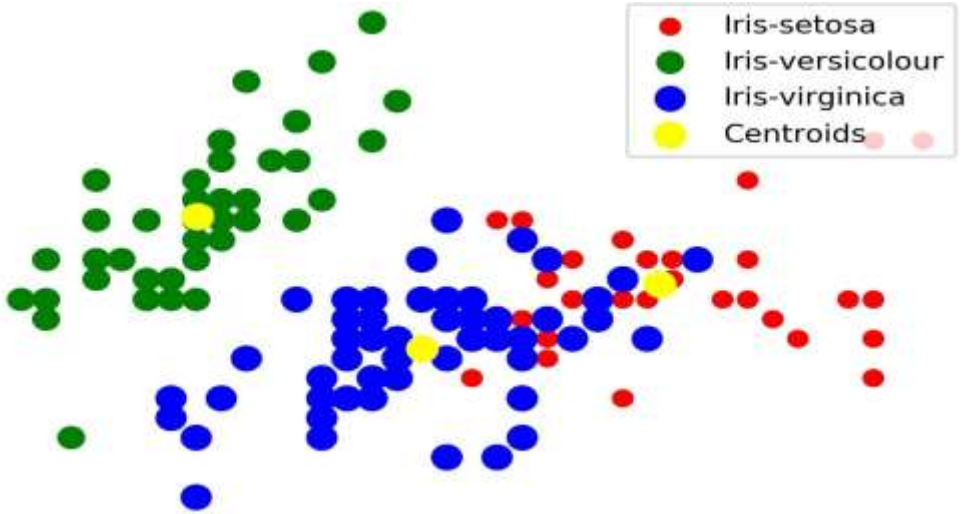
- **Clustering and Merging:**
- BIRCH employs a two-phase process:
- In the first phase, it scans the dataset and incrementally builds a CF tree.
- In the second phase, it applies a hierarchical clustering algorithm to the CFs to produce the final clusters.
- Threshold and Branching Factor:

- **BIRCH requires two parameters:**
- threshold: Maximum diameter of the subcluster.
- branching_factor: Maximum number of CFs in each node.
- These parameters influence the size and shape of the clusters.

BIRCH Clustering



BIRCH Clustering



BIRCH Clustering

- **from sklearn.cluster import Birch**
- **from sklearn.datasets import make_blobs**
- These lines import the necessary modules: Birch for BIRCH clustering and make_blobs to generate synthetic data for clustering.

- **blobs = make_blobs(n_samples=300, centers=3, cluster_std=1.0, random_state=42)**
- This line generates synthetic data with 300 samples, 3 centers, and a standard deviation of 1.0 for clustering.

- **birch = Birch(threshold=1.5, branching_factor=50, n_clusters=None)**
- This line initializes the BIRCH clustering algorithm with specific parameters:
- threshold: Maximum diameter of the subcluster.
- branching_factor: Maximum number of CFs in each node.
- n_clusters: Number of clusters. None means it's not fixed and determined automatically.

BIRCH Clustering

- **labels = birch.fit_predict(X)**
- This line fits the BIRCH algorithm to the data and assigns cluster labels to each data point.

- **import matplotlib.pyplot as plt**
- **plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis', marker='o', s=50, edgecolor='k')**
- **plt.title('BIRCH Clustering Result')**
- **plt.show()**
- This code segment visualizes the clustering result using a scatter plot, where different clusters are represented by different colors.