Hierarchical Clustering

Learn about hierarchical clustering via the agglomerative approach.

Chapter Goals:

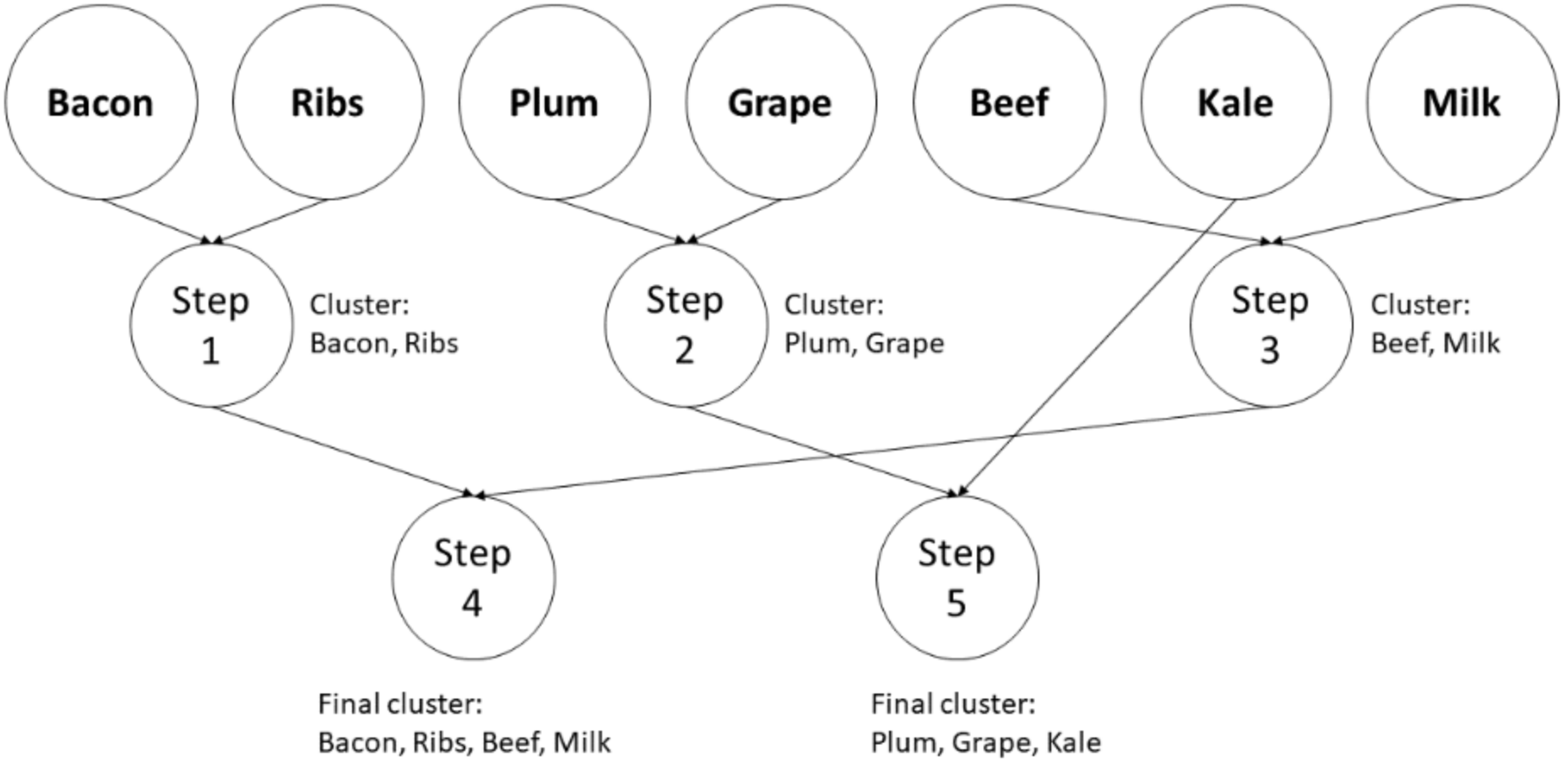
- Learn about hierarchical clustering using the agglomerative approach

A. K-means vs. hierarchical clustering

A major assumption that the K-means clustering algorithm makes is that the dataset consists of *spherical* (i.e. circular) clusters. With this assumption, the K-means algorithm will create clusters of data observations that are circular around the centroids. However, real life data often does not contain spherical clusters, meaning that K-means clustering might end up producing inaccurate clusters due to its assumption.

An alternative to K-means clustering is hierarchical clustering. Hierarchical clustering allows us to cluster any type of data, since it doesn't make any assumptions about the data or clusters.

There are two approaches to hierarchical clustering: bottom-up (divisive) and top-down (agglomerative). The divisive approach initially treats all the data as a single cluster, then repeatedly splits it into smaller clusters until we reach the desired number of clusters. The agglomerative approach initially treats each data observation as its own cluster, then repeatedly merges the two most similar clusters until we reach the desired number of clusters.



Agglomerative clustering for seven different food items into two final clusters. Each step the two most similar clusters are merged.

In practice, the agglomerative approach is more commonly used due to better algorithms for the approach. Therefore, we'll focus on using agglomerative clustering in this chapter.

B. Agglomerative clustering

In scikit-learn, agglomerative clustering is implemented through the `AgglomerativeClustering` object (part of the `cluster` module). Similar to the `KMeans` object from the previous chapter, we specify the number of clusters with the `n_clusters` keyword argument.

The code below demonstrates how to use the `AgglomerativeClustering` object (with 3 clusters).

```
from sklearn.cluster import AgglomerativeClustering
agg = AgglomerativeClustering(n_clusters=3)
# predefined data
agg.fit(data)

# cluster assignments
print('{}\n'.format(repr(agg.labels_)))
```

RUN SAVE RESET

Close

Output 2.098s

```
array([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, 2, 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,
       2, 2, 2, 0, 0, 2, 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 0, 2, 2, 2, 2,
       2, 0, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 0])
```

Since agglomerative clustering doesn't make use of centroids, there's no `cluster_centers_` attribute in the `AgglomerativeClustering` object. There's also no `predict` function for making cluster predictions on new data (since K-means clustering makes use of its final centroids for new data predictions).