In hierarchical clustering, this new step also consists of finding the optimal number of clusters. Only this time we’re not going to use the elbow method. We are going to use the dendrogram.

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset = pd.read\_csv('Mall\_Customers.csv')

X = dataset.iloc[:, [3,4]].values

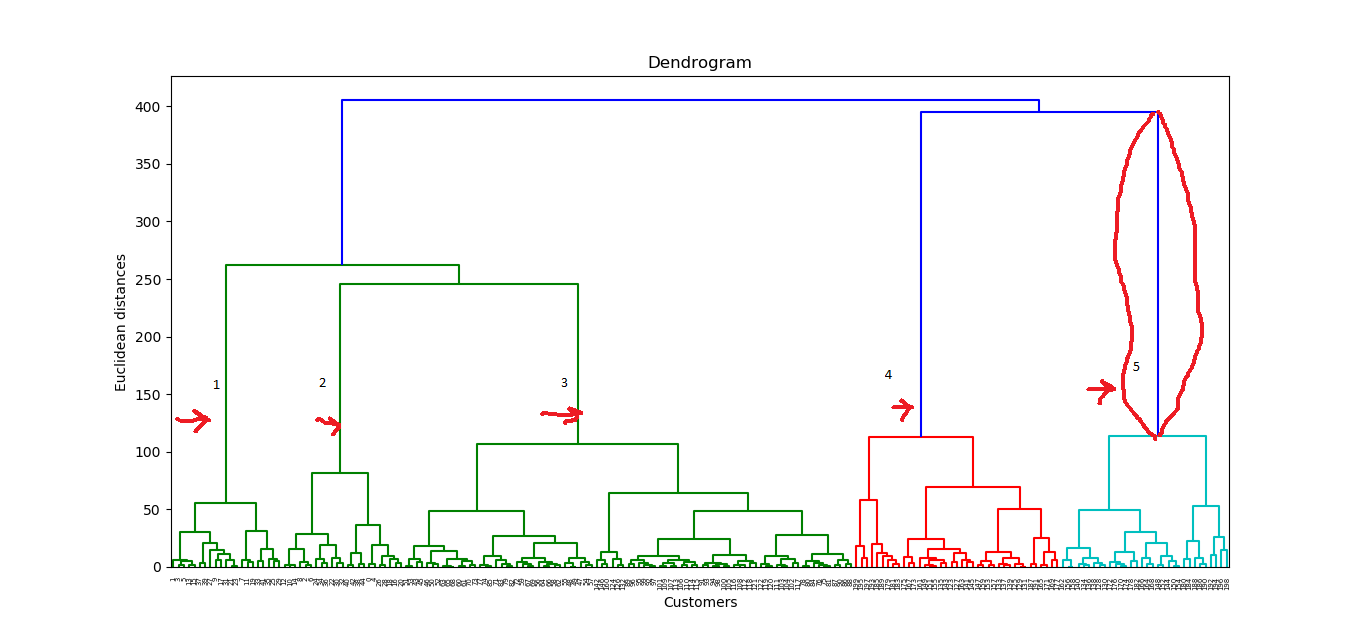
scipy.cluster.hierarchy as sch

dendrogram = sch.dendrogram(sch.linkage(X, method = "ward"))  
plt.title('Dendrogram')  
plt.xlabel('Customers')  
plt.ylabel('Euclidean distances')  
plt.show()

**Ward** method tries to minimize the variance within each cluster.

In K-means when we were trying to minimize the wcss to plot our elbow method chart, here it’s almost the same the only difference is that instead of minimizing wcss we are minimizing the within-cluster variants.

That is the variance within each cluster. Below is the dendrogram diagram.



The x-axis consists of the customers and y-axis consists of the Euclidean distance between the clusters.

How do we determine the optimal number of clusters from this diagram?

We look for the largest distance that we can vertically without crossing any horizontal line and this one is the red framed line on the above diagram.

Let's count the number of lines on the diagram and determine the optimal number of clusters.

Cluster number will be 5 for this dataset.

**#4 Fitting hierarchical clustering to the Mall\_Customes dataset  
# There are two algorithms for hierarchical clustering:**

**Agglomerative Hierarchical Clustering and Divisive Hierarchical Clustering.**

**We choose Euclidean distance and ward method for our algorithm class**

from sklearn.cluster import AgglomerativeClustering   
hc = AgglomerativeClustering(n\_clusters=5,

affinity='euclidean',linkage ='ward')

**Lets try to fit the hierarchical clustering algorithm to dataset X while creating the clusters vector that tells for each customer which cluster the customer belongs to.**

y\_hc=hc.fit\_predict(X)

Table

Description automatically generated

**Predicted Clusters for each of the mall customer**

**Visualizing the clusters.**

**This code is similar to k-means #visualization code.**

**We only replace the y\_kmeans vector name to #y\_hc for the hierarchical clustering**

plt.scatter(X[y\_hc==0, 0], X[y\_hc==0, 1], s=100, c='red', label ='Cluster 1')  
plt.scatter(X[y\_hc==1, 0], X[y\_hc==1, 1], s=100, c='blue', label ='Cluster 2')

plt.scatter(X[y\_hc==2, 0], X[y\_hc==2, 1], s=100, c='green', label ='Cluster 3')

plt.scatter(X[y\_hc==3, 0], X[y\_hc==3, 1], s=100, c='cyan', label ='Cluster 4')

plt.scatter(X[y\_hc==4, 0], X[y\_hc==4, 1], s=100, c='magenta', label ='Cluster 5')

plt.title('Clusters of Customers (Hierarchical Clustering Model)')  
plt.xlabel('Annual Income(k$)')  
plt.ylabel('Spending Score(1-100')  
plt.show()

Chart, scatter chart

Description automatically generated

**Cluster1(Red), Cluster2 (Blue), Cluster3(Green), Cluster4(Cyan), Cluster5 (Magenta)**

Let’s define these clusters as the customers' segment of a mall.

Table

Description automatically generated

That’s it for a typical Hierarchical Clustering Model. You can see the dataset and all the codes in the reference part.