**Hierarchical Clustering**

* Hierarchical Clustering permite crearea clusterilor specificându-se numărul lor sau distanța maximă dintre puncte
* Două abordări a algoritmului Hierarchical Clustering: de aglomerare și de dezbinare
* Abordarea de aglomerare (Agglomerative approach) – fiecare punct reprezintă un cluster și în funcție de similitudine are loc unificarea clusterelor
* Abordarea de dezbinare (Divisive approach) – toate punctele sunt incluse într-un cluster și în funcție de similitudinea aceste cluster se divizează
* Etapele algoritmului Hierarchical Clustering cu abordarea Agglomerative:
* Se compară toate punctele pentru a determina punctele cu un grad înalt de similitudine
* Se unesc aceste puncte într-un cluster
* Se compară din nou toate punctele inclusiv noul cluster și se formează un alt cluster nou
* Procedura se repetă până când se formează un cluster ce include toate punctele
* Similitudinea a 2 puncte se măsoară ca distanța dintre aceste puncte și poate fi: distanța Euclidean, Manhattan, Cosinus, etc
* Pentru determinarea distanței Euclidean este necesară scalarea valorilor caracteristicilor în gama 0...1 utilizând MinMaxScaler
* Dendograma este graficul ce permite afișarea potențialelor clustere fiind util la determinarea numărului acestora dar necesitând și multe resurse de calcul
* Parametrul linkage determină care puncte se utilizează la determinarea distantei dintre 2 clustere și poate avea valorile: Ward, Average, Minimum, Maximum
* Algoritmul de elaborare a modelului Hierarchical clustering în Scikit-Learn:
* importul datelor
* vizualizarea informației despre date ***df.info()***
* vizualizarea informației statistice despre df ***df.describe()***
* vizualizarea numărului de date in coloanele categoriale ***df['col'].value\_counts()***
* conversia datelor coloanei origin in valori numerice ***X = pd.get\_dummies(X)***
* importul modului de scalare MinMaxScaler ***from sklearn.preprocessing import MinMaxScaler***
* Crearea unui obiect de scalarea ***scaler = MinMaxScaler()***
* Scalarea datelor X ***X\_scalat = scaler.fit\_transform(X)***
* formarea unui df cu datele scalate ***df\_scalat=pd.DataFrame(X\_scalat,columns=X.columns)***
* Vizualizarea datelor cu ajutorul ***heatmap sns.heatmap(df\_scalat);***
* Elaborarea dendogramei doar pe linii cu ajutorul clustermap ***sns.clustermap(df\_scalat,col\_cluster=False)***
* importul modulului AgglomerativeClustering ***from sklearn.cluster import AgglomerativeClustering***
* crearea modelului pentru 4 clustere ***model = AgglomerativeClustering(n\_clusters=4)***
* Ajustarea modelului și determinarea clusterului fiecarei date ***cluster\_labels = model.fit\_predict(X\_scalat)***
* vizualizarea dependențelor din datele originale in functie de clustere ***sns.scatterplot(data=df,x=col1,y=col2,hue=cluster\_labels)***
* Vizualizarea numărului de clustere ***model.n\_clusters\_***
* Vizualizarea numărului de frunze in ierarhie ***model.n\_leaves\_***
* Vizualizarea punctelor de unificare in ierarhie ***model.children\_***
* Repetarea pașilor pentru modelul cu 2 clustere ***model = AgglomerativeClustering(n\_clusters=2)***
* Distanta minima la care punctele nu se vor uni in cluster ***dis\_min=0***
* Distanta maxima teoretica dintre puncte care va permite includerea tuturor punctelor într-un cluster ***dis\_max=np.sqrt(num\_caracteristici)***
* crearea modelului cu setarea 0 a distantei care nu mai unește puncte in cluster (fiecare punct va fi un cluster) ***model = AgglomerativeClustering(n\_clusters=None,distance\_threshold=0)***
* Ajustarea modelului și determinarea clusterului fiecărei date ***cluster\_labels = model.fit\_predict(X\_scalat)***
* importul modulelor scipy pentru determinarea distantelor într-o ierarhie ***from scipy.cluster.hierarchy import linkage***
* determinarea matricei cu distante ce include indiciile primului, celui de al doilea, distante dintre ele si numărul de puncte in clusterul format ***linkage\_matrix = linkage(model.children\_)***
* importul modulelor scipy pentru explorarea dendogramei ***from scipy.cluster.hierarchy import dendrogram***
* Elaborarea dendogramei ***hierarchy.dendrogram(linkage\_matrix)***
* trucarea dendogramei conform numărului de puncte frunze egal cu 48 ***hierarchy.dendrogram(linkage\_matrix,truncate\_mode='lastp',p=48)***
* trucarea dendogramei conform numărului nivele egal cu 5 ***hierarchy.dendrogram(linkage\_matrix,truncate\_mode='level',p=5)***
* crearea modelului cu setarea 2 a distantei care nu mai unește puncte in cluster model = ***AgglomerativeClustering(n\_clusters=None,distance\_threshold=2)***
* Ajustarea modelului și determinarea clusterului fiecărei date ***cluster\_labels = model.fit\_predict(X\_scalat)***
* Determinarea numărului de clustere ***nr\_cluster = model.n\_clusters\_***
* Determinarea matricea cu distante ***linkage\_matrix = linkage(model.children\_)***
* Elaborarea dendogramei trucate la numărul de clustere ***hierarchy.dendrogram(linkage\_matrix,truncate\_mode='lastp',p=nr\_cluster)***