TD4 – Partie A – Clustering avec Python

Classification avec Orange

1. Chargez le fichier « naissances_regions.xls » dans un dataframe nommé « naissances »
2. Affichez les premières lignes. Combien y'a-t-il de variables ? d'individus ?
3. Le dataframe représente des statistiques sur les naissances dans les régions du Burkina.

   Nous souhaitons réaliser un clustering de ces régions en trois groupes, mais avant, nous ferons une ACP pour percevoir la distribution des données.
4. Créez un objet « X » qui va contenir les données du dataframe naissances, avec les variables « naissances » , « prematures » et « moinsde25 » (qui désigne le nombre de naissances à moins de 2,5 kg). Nous omettons les deux autres statistiques car elles portent plus ou moins les mêmes informations que les trois précédentes.
5. Créez un objet « regions » qui va contenir la variable « region » transformée en liste.

   ```
   regions = naissances['region'].tolist()
   ```
6. Chargez le module correspondant, instanciez puis appliquez une ACP sur les données X. On mettra les projections des individus dans une variable « components ».

   ```
   from sklearn.decomposition import PCA
   pca = PCA(n_components=2)
   components = pca.fit_transform(X)
   ```
7. En utilisant le script suivant, affichez la projection des observations sur le plan factoriel.

   Est-ce que cette visualisation fait paraître des groupes ?

```
import matplotlib.pyplot as plt
fig = plt.figure(figsize = (15,15))
plt.scatter(components[:,0], components[:,1])

for i, reg in enumerate(regions):
    x = components[i,0] #comp1
    y = components[i,1] #comp2
    plt.annotate(reg, # this is the text
                 (x,y), # this is the point to label
                 textcoords="offset points", # how to position the text
                 xytext=(0,10), # distance from text to points (x,y)
                 ha='center') # horizontal alignment can be left, right or center

plt.grid()
```

8. Nous allons à présent réaliser la classification ascendante hiérarchique. Calculez la matrice des liens, et mettez-là dans un objet « Z » comme ci-dessous.

```
from scipy.cluster.hierarchy import linkage
Z = linkage(X, method='average', metric='euclidean') #Matrice des distances
```

9. En exploitant la matrice des liens Z, affichez le dendrogramme

```
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram
plt.title("Dendrogramme") #Affichage du dendrogramme pour visualiser
dendrogram(Z, labels=regions, orientation='right',color_threshold=12000)
plt.show()
```

10. Combien de groupes peut-on retenir si on se fie à la coloration du dendrogramme ?
11. Faites la classification en trois groupes. Affichez les groupes.

```
from scipy.cluster.hierarchy import fcluster
groupes_cah = fcluster(Z,t=3,criterion='maxclust') #Clustering en 3 classes
```

12. Refaites le clustering en 3 classes avec un Kmeans, puis affichez les groupes.
13. En utilisant l'indice de Rand ajusté, comparez les deux clustering proposés.
14. Modifiez le graphique de la question 8 à la 3è ligne de la façon suivante, de sorte à colorer selon les groupes trouvés par la CAH.

```
plt.scatter(components[:,0], components[:,1], c=groupes_cah)
```

15. Faites de même en coloriant selon les groupes trouvés par le kmeans. Quel clustering vous parait-il le plus approprié (visuellement) ?
16. Bonus : refaites tout ceci en utilisant l'outil Orange.