

|                     |                                                                                                                                                  |        |    |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|--------|----|
| Année universitaire | 2022-2023                                                                                                                                        |        |    |
| Département         | Informatique                                                                                                                                     | Niveau | L3 |
| Matière             | LIFDATA                                                                                                                                          |        |    |
| Enseignant          | Khalid Benabdeslem, H. Elghazel et M. Hennequin                                                                                                  |        |    |
| Intitulé TD/TP :    | Extraction de profils à partir de données                                                                                                        |        |    |
| Contenu             | <ul style="list-style-type: none"> <li>• Clustering (Classification non supervisée)</li> <li>• Évaluation de la qualité de partitions</li> </ul> |        |    |

## Remarque

Ce TP est à rendre sur Claroline. Il faudra rendre le notebook jupyter commenté. Ne pas oublier de mentionner vos noms et prénoms dans le nom du fichier qui sera sous la forme : **TP\_Clust\_A\_rendre\_Noms\_Prenoms.ipynb**.

## Préambule

Pour plus de détails concernant :

- le langage Python vous pouvez aller sur le site suivant : <http://www.python-course.eu/index.php>
- la librairie **Scikit-learn** vous pouvez aller sur le site suivant : <http://scikit-learn.org>

Pour lancer le notebook Python, il faut taper la commande **jupyter notebook** dans votre dossier de travail. Une fenêtre va se lancer dans votre navigateur pour ouvrir l'application **Jupyter**. Créer un nouveau notebook Python et taper le code suivant dans une nouvelle cellule :

```
import numpy as np
np.set_printoptions(threshold=10000,suppress=True)
import pandas as pd
import warnings
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')
```

La classification (les Anglo-saxons parlent de clustering) est l'opération statistique qui consiste à regrouper des objets (individus ou variables) en un nombre limité de groupes, les classes (ou segments, ou clusters) , qui ont deux propriétés. D'une part, ils ne sont pas prédéfinis par l'analyste mais découverte au cours de l'opération, contrairement aux classes du classement. D'autre part, les classes de la classification regroupent les objets ayant des caractéristiques similaires et séparent les objets ayant des caractéristiques différentes (homogénéité interne et hétérogénéité externe), ce qui peut être mesuré par des critères telle l'inertie interclasse et l'inertie intra-classe.

Nous allons étudier dans la suite deux approches de clustering (k-moyennes "**KMeans**" et la classification Ascendante Hiérarchique "**AgglomerativeClustering**" du package **sklearn.cluster**) que nous allons appliquer sur le jeu de données des **villes**.

## Questions

1. Appliquez la procédure **KMeans** sur ce jeu de données pour obtenir 3 clusters
  - Donner une visualisation graphique des villes projetées dans le plan principal. Les villes de chaque cluster devraient avoir une couleur différente des villes des autres clusters (voir code ci-dessous).

**X\_pca** étant la matrice des données transformées par l'**ACP**, **labels** étant le vecteur contenant le nom des instances (ici les villes), **clustering** étant le clustering obtenu.

```
colors = ['red','yellow','blue','pink']
plt.scatter(X_pca[:, 0], X_pca[:, 1], c= clustering,
            cmap=matplotlib.colors.ListedColormap(colors))
for label, x, y in zip(labels, X_pca[:, 0], X_pca[:, 1]):
    plt.annotate(label, xy=(x, y), xytext=(-0.2, 0.2), textcoords='offset points')
plt.show()
```

2. Appliquez la procédure **AgglomerativeClustering** sur ce jeu de données pour obtenir trois clusters avec différentes méthodes d'agrégation (il faut essayer **ward**, **average** et **single**).
  - Donner à chaque fois une visualisation graphique des villes projetées dans le plan principal. Les villes de chaque cluster devraient avoir une couleur différente des villes des autres clusters.

Nous allons maintenant déterminer la meilleure partition (nombre de clusters) pour les méthodes **KMeans** et **AgglomerativeClustering**. Pour cela, nous allons utiliser le critère "**Silhouette index**" (**metrics.silhouette\_score** de **scikit-learn**).

3. Utiliser cet indice dans une boucle de 5 itérations au maximum. Les 5 itérations correspondent aux 5 partitions possibles i.e. en 2, 3, 4, 5 et 6 classes issues de **KMeans**. Déduire la meilleure partition qui correspond à un indice maximal pour l'indice **Silhouette**.
4. Utiliser cet indice dans une boucle de 5 itérations au maximum. Les 5 itérations correspondent aux 5 partitions possibles i.e. en 2, 3, 4, 5 et 6 classes issues de **AgglomerativeClustering** (avec à chaque le critère d'agrégation **ward**, **average** et **single**). Déduire la meilleure partition qui correspond à un indice maximal pour l'indice **Silhouette**.
5. En considérant 3 clusters, quelle méthode donnera la meilleure partition ?
6. Citer les avantages et les inconvénients des méthodes de classification hiérarchiques (**AgglomerativeClustering**) et celles de partitionnement (**KMeans**).
7. Proposer dans une fonction une éventuelle approche permettant de combiner les points forts des méthodes hiérarchiques et des méthodes de partitionnement : on appelle une telle approche **mixte** ou **hybride**. Proposer une visualisation du nuage des villes et comparer les résultats obtenus avec ceux des questions précédentes.
8. Appliquer la fonction précédente sur les jeux de données "**wdbc.csv**" et "**spamb.csv**".