TP - Joueurs de rugby

Objectifs :

- ▷ Appliquer l'algorithme k-voisins

1 Travail à faire

On a relevé la taille et le poids des différents joueurs de rugby du *Top 14* ainsi que leur poste sur le terrain au cours de la saison 2023-2024. Vous allez travailler sur un tableau de données des joueurs de rugby du Top 14 issues de la page de la ligue nationale de rugby : https://top14.lnr.fr/joueurs

Votre travail consiste à attribuer un poste sur le terrain à un joueur à partir de son poids et de sa taille en utilisant l'**algorithme des k plus proches voisins** :

— Demi mêlée, arrière, 1ère ligne, 2ème ligne, ...,*etc.*

2 Exercices

Exercice 1 : *Extraction des données*

1. Analyser le fichier JoueurTop14.csv avec *Notepad++* par exemple.
2. Ecrire une fonction `liste_joueurs` de paramètre un fichier `csv` et qui renvoie une liste de dictionnaires contenant les données `'Poste'`, `'Taille (en cm)'` et `'Poids (en kg)'` du fichier.

Exercice 2 : *Liste des postes*

Ecrire une fonction `postes` qui a pour paramètre la liste des joueurs et qui renvoie un dictionnaire dont les clés sont les postes et les valeurs 0, plus tard cela correspondra au nombre de postes.

Exercice 3 : *Liste des joueurs*

Ecrire une fonction `liste_joueurs_par_rapport` qui a les paramètres `taille`, `poids`, `liste` des joueurs qui renvoie une liste de dictionnaires ayant deux clés le poste et la distance par rapport aux `'taille'` et `'poids'` saisis.

Exercice 4 : *KNN*

Ecrire une fonction `KNN` qui a les paramètres `k` le nombre de voisins, `taille`, `poids`, `liste_des_joueurs` qui renvoie le dictionnaire des postes mis à jour.

Exercice 5 : *Prédiction*

Ecrire une fonction `prediction` qui a les paramètres `taille`, `poids`, `k` le nombre de voisins, `liste_des_joueurs`, qui renvoie une prédiction sur le `'Poste'` le plus adapté.

Voici le code permettant d'afficher le nuage de points :

```
import matplotlib.pyplot as plt
def representation(data=joueurs):
    for i in range(len(data)):
        lacouleur="tab:"
        if(data[i]['Poste']=="Talonneur"):
            lacouleur+="blue"
            lemarker="x"
            label="Avant"
        elif(data[i]['Poste']=="Pilier"):
            lacouleur+="red"
            lemarker="+"
            label="Melee"
    elif(data[i]['Poste']=="2eme ligne"):
```

```
        lacouleur+="red"
        lemarker="+"
        label="2eme ligne"
    elif(data[i]['Poste']=="3eme ligne"):
        lacouleur+="green"
        lemarker="1"
        label="3eme ligne"
    elif(data[i]['Poste']=="Melee"):
        lacouleur+="purple"
        lemarker="."
        label="Melee"
    elif(data[i]['Poste']=="Ouverture"):
        lacouleur+="purple"
        lemarker="."
        label="Ouverture"
    elif(data[i]['Poste']=="Centre"):
        lacouleur+="brown"
        lemarker="*"
        label="Centre"
    elif(data[i]['Poste']=="Ailier"):
        lacouleur+="blue"
        lemarker="*"
        label="Ailier"
    else :
        lacouleur+="orange"
        lemarker="^"
        label="Arriere"
    plt.plot(int(data[i]['Poids']), int(data[i]['Taille']),
       color=lacouleur,marker=lemarker,label=label)
    plt.xlabel("Poids (en kg)")
    plt.ylabel("Taille (en cm)")
    plt.axis('equal') #pour avoir un repere orthonorme
    plt.show()
representation()
```

2.1 Avec un graphique

Voici le nuage de points de cette série réalisée avec le module `matplotlib`. Il faut être dans un repère orthonormé pour utiliser la distance euclidienne.