UNIVERSITÉ RENNES 2

Projet k-plus proches voisins

Xavier André & Romain Tavenard

1 Préambule

Pour ce projet, vous devrez travailler par groupes de 2 ou 3 et votre rendu se fera sous la forme d'un fichier Python. Ce fichier devra être nommé `P01.py` et contenir les noms et numéros étudiant de tous les membres du groupe commentés, en en-tête du fichier, comme dans l'exemple suivant :

```
# 22000002 Paul Machin
# 22000227 Yolène Truc

# Section 1 : les imports
# [...]
```

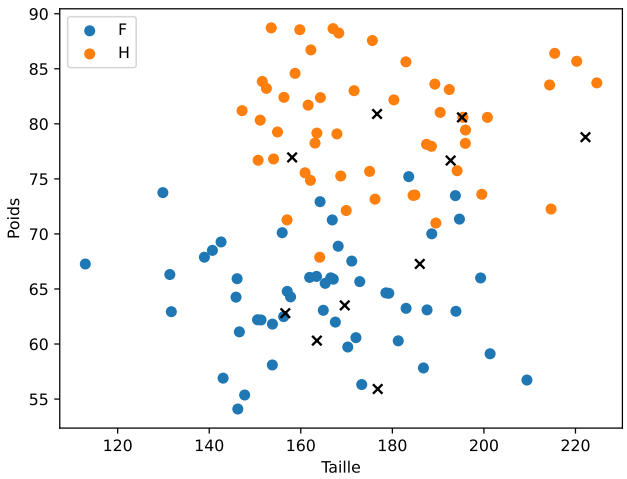
La date limite de rendu est indiquée sur CURSUS dans l'espace de dépôt.

2 Contexte

Pour ce projet, vous devrez implémenter un classifieur automatique. Un classifieur automatique est un modèle qui permet de prédire une caractéristique d'un individu statistique à partir d'attributs descriptifs de cet individu.

Cette prédiction se base sur des connaissances acquises grâce à un **jeu d'entraînement**. Un jeu d'entraînement est un ensemble d'individus pour lesquels on a accès à la fois à des attributs descriptifs et à la valeur de la caractéristique à prédire. En se basant sur ces individus, le but sera de prédire ladite caractéristique pour de nouveaux individus (contenus dans ce que l'on nomme un **jeu de test**) à partir de leurs attributs descriptifs uniquement.

Dans l'image ci-dessous, nous prenons l'exemple d'un problème de classification dans lequel on souhaite prédire le sexe d'un individu à partir de son poids et sa taille (notez que cet exemple est particulièrement inintéressant, mais il a au moins l'intérêt de se comprendre facilement) :



Sur cette image, le jeu d'entraînement est représenté par des points (orange ou bleu en fonction du sexe de l'individu) alors que le jeu de test est représenté par des croix noires (on n'a donc pas pour ces individus l'information de leur sexe). Le but d'un classifieur sera alors de prédire le sexe des individus du jeu de test à partir de son poids et de sa taille.

2.1 Les k-plus proches voisins

Vous étudierez lors de vos deux années de Master de nombreux modèles de classification supervisée qui peuvent permettre de fournir une solution pour ce problème. Dans ce projet, nous nous concentrerons sur les classifieurs dits "k-plus proches voisins".

Le principe de fonctionnement de ces classifieurs est relativement simple. Il s'agit, pour chaque individu X_i du jeu de test, de chercher parmi les individus du jeu d'entraînement les k individus qui sont les plus proches de X_i. Si une majorité de ces individus est de la classe `"F"` (sexe féminin), on prédira alors la classe `"F"` pour l'individu X_i, et sinon on prédira la classe `"M"`.

3 Code à produire

3.1 Préparation de votre projet Python

Pour ce projet, vous allez créer un nouveau sous-dossier `Projets` dans votre dossier `M1_S1_Python`, et ouvrir le dossier `Projets` dans VS Code (Menu Fichier -> Nouvelle fenêtre puis "Ouvrir un dossier"). Dans ce dossier `Projets`, vous allez télécharger le fichier `P01_utils.py` et créer un nouveau fichier `P01.py` dans lequel vous coderez.

3.2 Récupération et visualisation des données

1. En utilisant les fonctions `lire_donnees` et `visualiser_donnees` du module `P01_utils`, créez un jeu d'entraînement (constitué de 100 individus) et un jeu de test (constitué de 10 individus) et visualisez-les. *Pour la suite de l'énoncé, vous pourrez commenter l'appel à la fonction de visualisation pour que celle-ci ne se déclenche pas à chacun de vos tests.*

3.3 Implémentation des k-plus proches voisins en "pur Python"

Dans cette partie, vous tâcherez d'implémenter l'algorithme des k-plus proches voisins en "pur Python", c'est-à-dire sans utiliser (ou presque) les facilités de la librairie `numpy`.

1. Écrivez une fonction `dist` qui prend en entrée deux vecteurs `X_i` et `X_j` et retourne leur distance euclidienne.
2. Implémentez une fonction qui permette d'obtenir les indices des k plus proches voisin d'un individu de test parmi le jeu d'entraînement. Vous êtes autorisé(e) pour cela à utiliser la fonction `argsort` de la librairie `numpy`.
3. Implémentez une fonction qui, à partir d'une liste des classes des k-plus proches voisins (qui peut être `["F", "F", "H"]` par exemple si $k = 3$), calcule la classe la plus représentée dans la liste (ici `"F"`).
4. Implémentez une fonction `k_plus_proches_voisins_liste` qui prend en entrée un jeu de données d'entraînement, un jeu de données de test et un nombre k et retourne la liste des prédictions d'un modèle de k-plus proches voisins pour ces données. Si k n'est pas renseigné, il devra être fixé à 1 (seul le plus proche voisin est alors considéré).

3.4 Ré-implémentation des k-plus proches voisins en utilisant numpy

Dans cette partie, il vous est demandé de fournir une nouvelle implémentation de l'algorithme des k-plus proche voisin en utilisant, autant que faire se peut, les fonctions des modules `numpy` et `scipy`. Vous pourrez notamment utiliser les fonctions suivantes :

— `cdist` du module `scipy.spatial.distance`
— `argsort` du module `numpy`
— `sum` du module `numpy` (notez que la somme d'un vecteur de booléens correspond au nombre d'occurrences de la valeur `True`)

La fonction `k_plus_proches_voisins_numpy` que vous implémenterez dans cette partie aura la même signature que la fonction `k_plus_proches_voisins_liste` de la partie précédente, mais (si tout se passe bien :) son code devrait être beaucoup plus succinct.