



Devoir Surveillé d'informatique

1^{er} semestre AU : 2021 – 2022

Date : 21 Octobre 2021

Section : PB2

Nombre de page : 3

L'utilisation des calculatrices n'est pas autorisée pour cette épreuve.

Le langage de programmation sera obligatoirement Python.

*

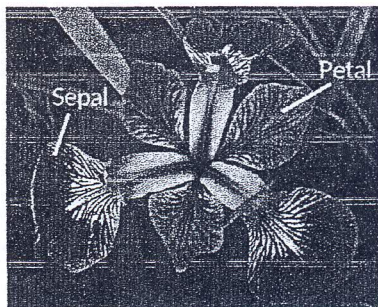
* *

Implémentation. Dans ce sujet, nous adopterons la syntaxe du langage Python. On rappelle qu'en Python, il importe de bien respecter les indentations car elles permettent de définir des blocs.

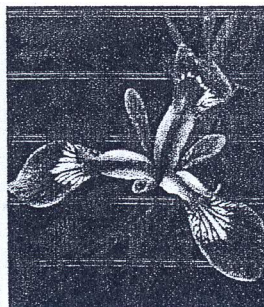
Fleurs IRIS :

L'iris est une fleur à la fois belle et élégante. Elle est également de culture facile au jardin ou en pot. La plantation et l'entretien sont des gestes qui permettront une belle floraison des iris de saison en saison et de faire durer les iris.

Dans ce sujet, nous allons nous servir du jeu de données Fleurs d'iris bien connu. Ce jeu de données comprend un total de 150 fleurs, réparties de manière égale entre les trois espèces de fleurs d'iris (setosa, virginica et versicolor).



Iris Versicolor



Iris Setosa



Iris Virginica

Quatre caractéristiques sont mesurées pour chaque fleur (c.-à-d. la longueur et la largeur du sépale et du pétale, en centimètres). Pour simplifier, nous nous basons dans ce problème sur la longueur et la largeur du sépale seulement.

Partie I. Représentation des fleurs

- Chaque fleur iris sera représentée sous forme d'une liste, par exemple :

`iris1 = [5.1 , 3.5 , 'setosa']` , où 5.1 est la longueur du sépale, 3.5 est la largeur du sépale et 'setosa' désigne l'espèce.

`iris2 = [7.0 , 3.2 , 'versicolor']`

`iris3 = [4.7 , 3.2 , 'setosa']`

- Les étiquettes des trois caractéristiques des fleurs seront stockées dans une liste :
`C = ['longueur du sépale', 'largeur du sépale', 'espèce']`
- La base de toutes les fleurs est une liste de liste de la forme :
`base_iris = [iris1, iris2, iris3, ...]` , où `iris1` est une liste qui représente une fleur

Question 1. Ecrire une fonction `get_iris_dict(C , iris)` qui crée et renvoie un dictionnaires D, où

- Les clés sont les éléments de la liste C
- Les valeurs sont les valeurs de chaque caractéristique de la fleur iris

Exemple pour la fleur iris I1, on aura un dictionnaire :

`D = {'longueur du sépale' : 5.1, 'largeur du sépale' : 3.5 , 'espèce' : 'Iris-setosa'}`

Question 2. Ecrire une fonction `get_all_iris(C , base_iris)` qui renvoie une liste de dictionnaire représentant les fleurs iris de la base.

Note : utiliser la fonction `get_iris_dict(C , iris)` de la question 1.

Partie 2 : Calcul de similarité des iris

Pour calculer la similarité des fleurs, on aura besoin de calculer une distance entre deux fleurs.

Question 3. Ecrire une fonction `distance(iris1 , iris2)` prenant en paramètres deux iris et renvoyant leur distance mutuelle.

Exemple :

Pour : `iris1 = [5.1 , 3.5 , 'setosa']`
`iris2 = [7.0 , 3.2 , 'versicolor']`

$$distance(iris1 , iris2) = \sqrt{(5.1 - 7.0)^2 + (3.5 - 3.2)^2}$$

Note : utiliser la fonction `sqrt` du module `math` pour calculer la racine carré.

Question 4.

Compléter la fonction `distance_all(iris_new , base_iris)` qui calcul la distance entre une nouvelle fleur `iris_new` et la liste des fleurs de notre base `base_iris`. Le résultat sera sous la forme d'une liste de tuple : (distance , espece)

```
def distance_all(iris_new , base_iris):  
    L_dist = list()  
    for iris in base_iris :  
        d = ...          # calcul de la distance entre iris et iris_new  
        tup = ( d , iris [ 2 ] )  
        ...              # ajout de tup dans L_dist  
    return L_dist
```

Partie 3 : (Classification)

Dans cette partie, nous allons travailler sur un algorithme d'apprentissage automatique afin de classer des nouvelles fleurs iris et de prédire leurs espèces. Cet algorithme d'apprentissage est assez simple à appréhender : l'algorithme des "k plus proches voisins" (en anglais "k nearest neighbors" : KNN).

Question 5.

Ecrire une fonction **KNN (iris_new , base_iris , k = 3)** qui renvoie les k plus proches voisins d'une nouvelle fleur iris en suivant cet algorithme :

1. Création de la liste des tuples des distances. (Utiliser la fonction **distance_all** de la question 4)
2. Trier dans l'ordre croissant la liste des distance (utiliser la fonction prédéfinie **sorted**)
3. Récupérer les k premiers iris de la liste ainsi triée.

Question 6.

Compléter la fonction **classification (iris_new , base_iris , K)** qui affiche la probabilité de l'espèce que l'on peut affecter à cette nouvelle fleur **iris_new** par le principe KNN.

```
def classification ( iris_new , base_iris , k = 3 ) :  
  
    L_iris_voisin = ...          # récupérer les k plus proches voisins de iris_new  
  
    L_espece_voisin = list()  
  
    for iris in L_iris_voisin :  
  
        ...                    # ajouter l'espèce de iris dans la liste L_espece_voisin  
  
    especes = ['setosa', 'versicolor', 'virginica']  
  
    for e in especes :  
  
        occ = ...              # calculer l'occurrence de e dans L_espece_voisin  
  
        print("la probabilité d'être dans l'espèce ", e , " = ", occ*100/k , "%")
```