

ROUX
Anthony

DS Informatique

TPSI 2.

Q1 SELECT idpatient FROM MEDICAL WHERE etat = 'lame dorsale';

Q2 SELECT nom, prenom FROM PATIENT
JOIN MEDICAL ON PATIENT.id = idpatient
WHERE etat = 'spondylolisthésis';

Q3 SELECT etat, count(*) FROM MEDICAL
GROUP BY etat;

Distinct

Q4 : Tableau N lignes, n colonnes sur 32 bits.

Pour des tableaux de grande taille, la bibliothèque numpy permet de faire du calcul matriciel et ainsi d'accéder rapidement aux données : par ligne ou colonne.

Q5 32 Bits correspondent à 4 octets.

Data Tableau : $N \times n \times 4 = 100\,000 \times 6 \times 4 = 2\,400\,000$ octets
 $= 2,4 \text{ Mo}$

Etat Vecteur : $N \times 4 = 100\,000 \times 4 = 400\,000 = 0,4 \text{ Mo}$.

Q6 def separationParGroupe (data, etat) :

Q7

(ARGS 1): (6, 6, i)

(ARGS 2): ('incidence-bassin', 'orientation-bassin', marker = 'o')

(ARGS 3): (data)

TEST: $i \neq j$

Q8 : Les diagrammes de la diagonale donnent une représentation globale de la proportion de patients en fonction du degré d'incidence au bassin. ✓

Ces hors-diagonale permettent de relier incidence et l'orientation du bassin ainsi que l'angle de lordose lombaire selon les états du patient soit : normal, hernie discale ou spondylolisthème. ✓

Q9 x_i vecteur colonne : de data.

$$0 \leq x_{normj} \leq 1.$$

Si $x_j = \min(x)$ alors $x_{normj} = 0$.
Si $x_j = \max(x)$ alors $x_{normj} = 1$.
Si $x_j \neq \min(x)$ et $x_j \neq \max(x)$ alors $x_{normj} \in]0;1[$.

Q10 def min_max(X) :

max, min = X[0], X[0]
for i in range(1, len(X)):
 if X[i] > max:
 max = X[i]
 if X[i] < min:
 min = X[i]
return max, min.

Il faut donner
une relation

Et la
complexité ?

Q11 def distance(z, data) :

for i in range(len(data)):
 return abs(z - i).

A revoir.

Q12 : T liste de : listes à 2 éléments



dist : représente les distances euclidiennes entre le n-uplet z
et chaque n-uplet x du tableau data.



de partie 1 ajoute à la liste T les distances euclidiennes à
leur position respective (position i) : ÉTAPE de création de T.

Partie 2 : on ajoute 1 à la valeur de chaque état,
opère multiplication par 3.
si n'y a donc plus d'état portant le numéro 0.

Partie 3 : Permet de renvoyer pour chaque élément de T, le
nombre de plus proches voisins k.

Q14 : la plupart du temps l'algorithme fonctionne
(dans plus de 70% des cas).

Cependant, le taux de réussite reste toujours inférieur à 75%
1 fois 4 au moins, l'algorithme échoue quelque que soit
la valeur de K. Efficacité moyenne.

Q15 def moyenne(x):
 may = 0
 for i in range(len(x)):
 may = may + x[i]
 return $\frac{\text{may}}{\text{len}(x)}$

def variance(x):
 var = 0
 for i in range(len(x)):
 var = var + (x[i] - (moyenne(x)))**2
 return $\frac{\text{var}}{\text{len}(x)}$

Q16

```
def synthese(data, etat):  
    for i in range(len(data)):  
        for j in range(len(etat)):  
            L = [moyenne(x[j]), variance(x[j])]  
    return L
```

A indexer et
initialiser

Q17

```
def fonction-gaussienne(a, moy, v):  
    return (1/(2*pi*v)**0.5)  
    * exp(-(a-moy)**2/(2*v))
```

Q18

```
def probabiliteGroupe(z, data, etat):
```

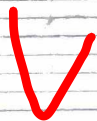
Q19

```
def prediction(p0, p1, p2):  
    if (p0 >= p1) and p0 >= p2:  
        return p0  
    elif p1 >= p2:  
        return p1  
    else:  
        return p2.
```

Consignes
pas
respectées

Q20 . Comme $P(X_i = z_i | Y = y_j)$ s'écrit selon une exponentielle, l'utilisation du logarithme permet la simplification des calculs dans l'algorithme.

(En effet le logarithme d'exponentielle x simplifie l'expression à x par ex.).



Q21 méthode KNN pour $K = 8$.

Méthode naïve bayésienne.

$\begin{pmatrix} 23 & 9 & 8 \\ 9 & 10 & 1 \\ 10 & 1 & 49 \end{pmatrix}$, matrice de confusion.

