KALMAR Gaëlle MPSI 1

DS 9

18/06/2020

Q1- SELECT id FROM MEDICAL
WHERE etat = " hernie discale "

Q2- SELECT nom, prenom FROM PATIENT, MEDICAL
WHERE PATIENT.id = MEDICAL.id
AND etat = " spondylolisthésis "

I "prefer" les jointures.

Q3- SELECT etat, COUNT(*) FROM MEDICAL
GROUP BY etat

Q4- Grâce à Numpy on peut utiliser directement les fonctions de calcul vectoriel ce qui est pratique pour de grands tableaux.

Q5- D'après l'énoncé les informations dans le tableau data sont codés sur 32 bits donc 4 octets pour N = 100 000, data prend donc $4 \times n \times N = 4 \times 6 \times 10^5 = 2,4$ Mo

Le vecteur etat prend $1 \times N = 10^5$ octets $= 0,1$ Mo (codé sur 8 bits)

Donc toutes ses informations prennent 2,5 Mo

Q6 -

```
def separationParGroupe(data, etat):
    st_1, st_2, st_0 = [], [], []
    for i in range(N):
        infos = data[i, :]
        if etat[i] == 0:
            st_0.append(infos)
        elif etat[i] == 1:
            st_1.append(infos)
        else:
            st_2.append(infos)
    return [st_0, st_1, st_2]
```

~~def info_Patients(m):~~
~~info = []~~
~~for j in range(h):~~
~~info.append(data[m, j])~~
~~return info~~

Q7 -

```
ARGS1 = (6, 6, 6*i + j)
ARGS2 = (groupe[k, j], groupes[k, i], marker = mark[k])
ARGS3 = (data[i])
TEST = if i != j
```

Q8 - Les digrammes de la diagonale permettent pour chaque attribut d'identifier la tranche de valeur moyenne.

Q 9- $x_{norm_j} = \dfrac{x_j - \min(X)}{\max(X) - \min(X)}$

Ainsi si on définit $f : x_. \longmapsto \dfrac{x - \min(X)}{\max(X) - \min(X)}$

On a bien $f(\min(X)) = 0$ et $f(\max(X)) = 1$

Q 10-

```
def min_max (X):
    if X[0] > X[1]:
        X_max = X[0]
        X_min = X[1]
    else:
        X_max =
```

```
def min_max (X):
    X_max = X[0]
    X_min = X[0]
    for i in range (len(X)):
        if X[i] > X_max :
            X_max = X[i]
        elif X[i] < X_min:
            X_min = X[i]

    return X_min, X_max.
```

Q 11-

```
def distance (z, data ):
    dist = []

    for i in range (N):
        ligne, l = data [i, :], []
        for j in range (n):
            l.append ( abs(ligne [j] - z [j] ))
```

```
    dist.append(l)
return dist
```

Q12- Partie 1 : Crée une liste (dist) de listes contenant le numéro de la ligne (i) et les distances euclidiennes entre le vecteur z et le $i^{ème}$ n-uplet x_i de data. Puis trie cette liste

Partie 2 : Crée une liste (select) de dimension 3 correspondant aux 3 états.
On regarde l'état des K plus proches voisins et on l'enregistre dans select

Partie 3 : Renvoie l'indice k de select dont select[k] est maximum.
Cela correspond à l'état auquel appartient le patient représentées par le vecteur z dont les informations sont

Q13- Les chiffres sur la diagonale représente le nombre de patient dont la prédiction correspondait bien à son état.

1ère ligne : Patient d'état 0 :
23 ont été prédit état 0
4 d'état 1 et 7 d'état 2

1ère colonne : Patient prédit d'état 0

23 était bien d'état 0
7 était d'état 1 et 5 d'état 2.

Cette matrice sert à identifier le nombre de bonnes prédictions : c'est la somme des nombres de la diagonale
Les autres sont des erreurs de prédiction.

Q14- L'algorithme n'est pas très efficace : en moyenne seulement 72% de bonne prédiction.

Q21- KNN: Pourcentage de réussite : $\frac{23+11+40}{100} = 74\%$
Méthode naïve bayésienne : $\frac{23+10+49}{100} = 82\%$

La 2ème méthode est plus efficace.