

HAMADI

Kalima

## DS: Informatique

1- SELECT idpatient FROM MEDICAL WHERE etat = 'hémor  
dysale';

2- SELECT nom: PATIENT, prenom: PATIENT, etat: MEDICAL FROM  
PATIENT JOIN MEDICAL ON PATIENT.id = MEDICAL.idpatient WHERE  
etat = 'spondylolisthesis';

Distinct

3- SELECT etat, COUNT(idpatient) FROM MEDICAL GROUP BY etat

4- On peut facilement extraire des lignes et obtenir un tableau plus  
petit. La bibliothèque numpy est plus efficace pour gérer les tableaux que  
Python (\*)

6- def reparationParGroup (data, etat):

etat 0, etat 1, etat 2 = [], [], []

for k in range (len (data)):

if etat k == 0:

etat 0.append (data [k])

elif etat [k] == 1:

etat 1.append (data [k])

else:

etat 2.append (data [k])

return etat 0, etat 1, etat 2.

Pas uniquement

7- ~~ARGS1 =  $n^2$~~

ARGS2 sert à représenter les trois sous-groupes et fonction de  $k$ , donc  $ARGS2 = (\text{groupes}[i], \text{groupe}[j], \text{mark} = k)$

L'histogramme de ARGS3 compte le nombre de données.

donc  $ARGS3 = \text{data}[k]$

TEST = ' $j \neq i$ '

8- Les diagrammes de la diagonale permettent de voir la valeur moyenne des données.

Les diagrammes hors diagonale permettent de faire des comparaisons selon la répartition des points.

9- ~~$$x_{\text{moy}} = \frac{\max(X) - \min(X)}{2j}$$~~

A plus détailler

10- def min\_max(X):

min = X[0]

max = X[0]

for k in X:

if k < min:

min = k

elif k > max:

max = k

return min, max

Il faut aussi justifier la complexité

11- def distance(z, data):

d = 0

for k in range(len(z))

d = sqrt((z[k] - data[k])\*\*2)

return d



12- Partie 1 : on ajoute un tableau contenant la distance  $d[i]$  et  $i$  à une liste  $T$ , puis on la trie. ✓

Partie 2 : select represent.

13- La diagonale de la matrice nous indique le taux de succès de l'algorithme. ✓

14- L'algorithme n'est pas efficace pour les petites valeurs (entre 0 et 5) et pour les grandes valeurs (entre 15 et 20). ✓

15- def moyenne (x):

$s = 0$

for  $k$  in  $x$ :

$s += k$

return  $(s / \text{len}(x))$

def variance(x):

$s = 0$

$m = \text{moyenne}(x)$

for  $k$  in  $x$ :

$s += k ** 2$

```
return ((s / len(y)) - m * + 2)
```

```
16- def synthex (data, etat):
```

```
    m = moyenne (x)
```

```
    v = variance (x)
```

```
    for k in range (len (data)):
```

```
        if etat == 0:
```

```
            if etat == 1:
```

```
            if etat == 2:
```

```
    return ([m, v])
```

```
17- def gaussienne (a, moy, v):
```

```
    moy = moyenne (X)
```

```
    v = variance (X)
```

```
    p = []  
    for a in range (len (X)):
```

```
    return p
```

```
18- def probabiliteGroupe (g, data, etat):
```

```
    return p
```

```
19- def prediction (
```

(\*) 5.  $N = 100\,000$  ligne,  $m = 6$  colonnes

code sur 32 bits = 4 octet. donc la quantité de mémoire est :

2,4 Mo