DS info

2. Analyse des données

Compliqué

1. SELECT idpatient FROM (SELECT * FROM PATIENT JOIN MEDICAL ON PATIENT.id = MEDICAL.idpatient) WHERE etat = "hernie discale";

2. SELECT nom, prenom FROM (SELECT * FROM PATIENT JOIN MEDICAL ON PATIENT.id = MEDICAL.idpatient) WHERE etat = "spondylolisthesis";

3.

X

Pas l'argument le plus pertinent

4. C'est plus lisible car sans Jane away, on verra les tableaux sans Jane matricielle!

5. Le tableau data contient 100 000 lignes et 6 colonnes chaque case codée sur 32 bits c'est 4 octets ainsi *

6.
```
def separationParGroupe(data, etat):
    A = []
    B = []
    C = []
    for i in range(len(data)):
        if etat[i][0] == 0:
            A.append(data[i])
        elif etat[i][0] == 1:
            B.append(data[i])
        else:
            C.append(data[i])
    return [A, B, C].
```

7. La double boucle permet de dessiner indépendamment chaque graphique de coo. (i,j) pour $i,j \in [\![0, n-1]\!]$ ainsi on commence par sélectionner le graphique correspondant dans la matrice à l'aide de la ligne `ax1 = plt.subplot(n, n, i*n+j+1)`

Ainsi `ARGS1 = n, n, i*n+j+1`.

3. Apprentissage et prédiction

9. on aurait $x_{norm\,j} = \frac{x_j - \min(x)}{\max(x) - \min(x)}$

10. def min-max (x):
    a = len x [0]
    b = len x [0]
    for i in range (len(x)):
        if len x [i] >= b:
            b = len x [i]
        elif len x [i] <= a:
            a = len x [i]
    return a, b

Il faut le justifier

on a bien une complexité linéaire.

11. def distance (x, data):
    A = []
    for i in range (len (data [0])):
        A.append (data
    for i in range (len (data [0])):
        A.append (data

* il requiert au total N × m × 4 octets soit 2,4 Mo. De même, état requiert 0,1 Mo ainsi les deux requièrent donc 2,5 Mo.

12. • la partie 1 crée une liste T et la trie.
• partie 2: créer une liste de 0
• partie 3: ça manque de détail

14. On remarque que l'algorithme est plutôt efficace mais le pourcentage ne dépasse pas 75% ainsi il n'est quand même pas totalement fiable.

V