

LEBRAT

Felix

NPS17

DS n°9 d'info

Q7) select idpatient from medical where etat="heute ~~déjà~~"

Q2) select nom, prenom from patient join medical on
patient.id = medical.id where etat="spondylolisthesis"

Q3) select etat, count(*) from medical group by etat

Q8) taille en octets : $N \times N \times 4 + N$
 $= 700\,000 \times 25$
 $= 2\,500\,000$

soit $\geq 5\text{ Mo}$

Q6) def separerParGroupe(data, etat) :

ret = [[], [], []]

for i in range(N) :

ret[etat[i]].append(data[i])

return ret

Q7) ARG1 = (n, n, i * n + j + 1)

ARG2 : (groupes[h][i:j], groupes[h][i:i], mark[h])

ARG3 : (groupes[i][i:i])

TEST : (i != j)

Q9) $z_{norm_i} = (x_i - \min(x)) / (\max(x) - \min(x))$

Q77) def distance(z, data):

```
    ret = []
    mm = [min-max(data[i:j]) for j in range(n)]
    for x in data:
        d = 0
        for i in range(n):
            d += ((x[i] - z[i]) / (mm[i][1] - mm[i][0])) ** 2
        ret.append(d ** (1/2))
    return ret
```

Q78) def min-max(x):

```
    min = x[0]
    max = x[0]
    for x in x:
        if x < min:
            min = x
        if x > max:
            max = x
    return (min, max)
```

Q79) La partie 1 construit la liste T de listes à deux éléments contenant : - la distance entre z et x pour x dans data

- l'indice de x dans data

et trie cette liste (dist contient les distances)

La partie 2 construit la liste select, indexée par les états, et qui indique pour chaque état, combien de n-uplet correspondant à cet état sont présents dans les k plus proches de z

La partie 3 cherche l'indice de l'état le plus représenté parmi les k plus proches n-uplets de z,

il est stocké dans ind

Q73) Les valeurs de la diagonale de la matrice
sont, pour chaque état, le nombre de fois
où l'algorithme a fait le bon diagnostic
ainsi, la trace de la matrice est le total
de diagnostics corrects.

Les valeurs de la première ligne correspondent
à l'état prédit lorsque l'état réel était 0
on en déduit qu'il y a eu $7+4$ erreurs lorsque
l'état réel était 0

De même, les valeurs de la 7ème colonne correspondent
aux états réels, sachant que l'état prédit est 0
ainsi, si l'état prédit est 0, il y a $\frac{7+5}{23+7+5}$

chances que ce soit une erreur.

Q74) L'algorithme ne dépasse pas 75% de réussite
le taux de réussite maximum est réalisé pour
 k autour de 70

Q75) def moyenne(x):

 somme = 0

 for x_i in x :

 somme += x_i

 return somme / len(x)

def variance(x):

 somme = 0

 moy = moyenne(x)

 for x_i in x :

 somme += $(x_i - moy) * x_i$

```
return somme / len(z)
```

```
Q76) def synthese (data, etat):
```

```
    groupes = separationParGroupe(data, etat)
```

```
    ret = []  
    for i in range(3):
```

```
        ret.append([])
```

```
        for j in range(6):
```

```
            ret[i].append (moyenne(groupe[i][j]),  
                            variance(groupe[i][j])))
```

```
    return ret
```

```
Q77) def gaussienne (a, moy, v):
```

```
    return exp  $\left( - (a - moy)^2 / (2 \times v) \right) / \sqrt{2 \pi \times v}$ 
```

```
Q78) def probabiliteGroupe (z, data, etat):
```

```
    syn = synthese (data, etat)
```

```
    groupes = separationParGroupe (data, etat)
```

```
    ret = []
```

```
    for i in range(3):
```

```
        p = 1
```

```
        for j in range(6):
```

```
            p *= gaussienne(z[j], syn[i][j][0],  
                             syn[i][j][1])
```

```
    p *= len(groupe[i]) / len(data)
```

```
    ret.append(p)
```

```
    return ret
```

Q 19) def prediction (z, data, etat):

p = probabilité_groupe (z, data, etat)

id = 0

for i in range(7, 3):

if p[i] > p[id]:

id = i

return id

les arguments z, data et etat sont les mêmes
que pour probabilité_groupe

Q 20) l'utilisation du logarithme permet d'obtenir
des données moins rapprochées les unes des autres,
ce qui rend la représentation sur un graphique
plus claire.

Q 21) méthode KNN: $\text{tr} \begin{pmatrix} 23 & 4 & 7 \\ 7 & 7 & 7 \\ 5 & 2 & 40 \end{pmatrix} = 74\%$ de réussite

méthode naïve bayésienne:

$\text{tr} \begin{pmatrix} 23 & 9 & 8 \\ 9 & 70 & 7 \\ 70 & 7 & 49 \end{pmatrix} = 82\%$ de réussite

ainsi, sur l'exemple traité, la méthode bayésienne
est plus efficace.