B05902091 黎峻碩

No:1 2 4 5 P1

7 8 9 11 12 P2

13 14 15 P3

Code P4

1. [c]

$h_w(X)$	w
+ + + +	(1, 1, 1, 3)
+ + + −	(196, -60, -28, 68)
+ + − +	(111, -125, -19, 121)
+ + − −	(30, -28, -15, 32)
+ − + +	(-3, 13, -19, 7)
+ − + −	(-39, 125, -137, 21)
+ − − +	(-39, 13, -37, 25)
+ − − −	(3, -3, -4, 3)
− + + +	(-3, 3, 4, -3)
− + + −	(43, -15, 38, -25)
− + − +	(43, -135, 146, -21)
− + − −	(4, -14, 19, -6)
− − + +	(-30, 28, 15, -32)
− − + −	(-110, 124, 19, -120)
− − − +	(-180, 58, 27, -66)
− − − −	(0, -1, -1, -1)

(4, -14, 19, -6)
− − + + (-30, 28, 15, -32)
− − + − (-110, 124, 19, -120)
− − − + (-180, 58, 27, -66)
− − − − (0, -1, -1, -1)

2. [d]

The perceptrons can be considered as two rays with the value of its x, y axis. For each ray, there is at most 2N ways to divide and then all + or all − would be overlaped once. Therefore it is 4N − 2.

4. [b]

Can be considered as the interval of radius, between 2 circles. Which is same as the growth function of intervals.

5. [b]

As it can be considered as the growth function of intervals, the VC dimension is also the same.

7. [d]

$\because$ H is a binary classifer with M hypothesis

$$M \leq 2^N$$

$$\log_2 M \leq N$$

8. [d]

$\because$ For all input, there would be at most $k+1$ results

i.e. Value depend on the number of 1's

for $k = 3$, there would be 0 to 3 1's in input

$\therefore$ The VC dimension would be $k+1$.

9. [c]

Necessary conditions:

- Some set of d distinct inputs is shattered by H.
- some set of $d+1$ distinct inputs is not shattered by H.
- any set of $d+1$ distinct inputs is not shattered by H.

11. [d]

$$E_{out}(h,\tau) = \underbrace{E_{out}(h,0) \cdot (1-\tau)}_{\text{error label correct}} + \underbrace{(1-E_{out}(h,0)) \cdot \tau}_{\text{correct label error}}$$

$$E_{out}(h,\tau) = E_{out}(h,0) \cdot (1-2\tau) + \tau$$

$$E_{out}(h,0) = \frac{E_{out}(h,\tau) - \tau}{1-2\tau}$$

$$E_{out}(h,0) = \frac{E_{out}(h,\tau) - \tau}{1-2\tau}$$

12. [b]

Assume that $P(f(x)=1) = \frac{1}{3}$, $P(f(x)=2) = \frac{1}{3}$, $P(f(x)=3) = \frac{1}{3}$

$P(y=1|f(x)=1) = 0.7$ $\quad$ $P(y=1|f(x)=2) = 0.2$ $\quad$ $P(y=1|f(x)=3) = 0.1$

$P(y=2|f(x)=1) = 0.1$ $\quad$ $P(y=2|f(x)=2) = 0.7$ $\quad$ $P(y=2|f(x)=3) = 0.2$

$P(y=3|f(x)=1) = 0.2$ $\quad$ $P(y=3|f(x)=2) = 0.1$ $\quad$ $P(y=3|f(x)=3) = 0.7$

$$\tilde{y} = \begin{cases} 1 & \text{avg. err} = \frac{1}{3}[(1-2)^2 \cdot 0.1 + (1-3)^2 \cdot 0.2] = 0.3 \\ 2 & \text{avg. err} = \frac{1}{3}[(2-1)^2 \cdot 0.2 + (2-3)^2 \cdot 0.1] = 0.1 \\ 3 & \text{avg. err} = \frac{1}{3}[(3-1)^2 \cdot 0.1 + (3-2)^2 \cdot 0.2] = 0.2 \end{cases}$$

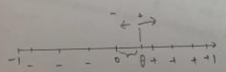
$\therefore E_{out}(f) = 0.6$.

14. [d]

N	δ
6000	26.548
8000	2.9056
10000	0.298
12000	0.02937
14000	0.0028

15. [b]

The error should be the distance between 0 and θ



$\therefore E_{out}(h_{+1,\theta}, 0)$ should be $\frac{|\theta|}{1-(-1)} = \frac{1}{2}|\theta|$

13. [b].

$\frac{1}{3}[1-(1\cdot0.7+2\cdot0.1+3\cdot0.2)]^2 + \frac{1}{3}[2-(1\cdot0.2+2\cdot0.7+3\cdot0.1)]^2 + \frac{1}{3}[3-(1\cdot0.1+2\cdot0.2+3\cdot0.7)]^2$

$= \frac{1}{3}(0.25+0.01+0.16)$

$= 0.14$

```python
import numpy as np
import random
import statistics

tau_list = [0, 0, 0.1, 0.1, 0.1]
N_list = [2, 20, 2, 20, 200]
repeat_time = 10000
s = [-1, 1]

sample_size = 100000
for ex_no in range(0, 5):
    N = N_list[ex_no]
    tau = tau_list[ex_no]
    diff_list = []

    for re in range(0, repeat_time):
        i = 0
        x1 = []
        exist = []
        while i < N:
            randn = random.uniform(-1, 1)
            if randn not in exist:
                exist.append(randn)

                change_sign = random.uniform(0, 1)
                sign = 1
                if change_sign < tau:
                    sign = -1

                if randn > 0:
                    y = 1*sign
                else:
                    y = -1*sign

                x1.append((randn, y))
                i = i + 1

        x1.sort(key=lambda x:x[0])
        theta = -1
        bestTheta = -1
        bestEin = 1
        bestS = -1

        #Theta = -1
        for s1 in s:
            Ein = 0
            for point in x1:
                if(point[0]-theta) > 0:
                    sign = 1*s1
                else:
                    sign = -1*s1
                if sign != point[1]:
                    Ein = Ein + 1

            if Ein/N < bestEin:
                bestEin = Ein/N
                bestS = s1
                bestTheta = -1

        #Theta = 0.5(xi+x(i+1))
        for i in range(0, N-1):
            theta = (x1[i][0]+x1[i+1][0])/2
            for s1 in s:
                Ein = 0
                for point in x1:
                    if(point[0]-theta) > 0:
                        sign = 1*s1
                    else:
                        sign = -1*s1
                    if sign != point[1]:
                        Ein = Ein + 1

                if Ein/N < bestEin:
                    bestEin = Ein/N
                    bestS = s1
                    bestTheta = theta

        Eout = 0.5*abs(bestTheta)*(1-tau) + (1-0.5*abs(bestTheta))*tau

        diff_list.append(Eout-bestEin)

    print(str(16+ex_no) + ' result: ')
    print(statistics.mean(diff_list))
```