Machine Learning HW2

R04922034 吳軒衡

1.

P (x, y) = P (X) P (y|x)
The probability of error that h makes in approximating noisy target y is
a). Makes an error when f is right
b). Don't make error when f is wrong
=> u*lambda + (1-u)(1-lambda)
= u*lambda + 1 - u - lambda + u*lambda
= 1 - u - lambda+ 2*u*lambda

2.

For h to be independent of u
Let lambda = 0.5
h = 1 - u - lambda + 2 * u*0.5
= 1- lambda

3.

```
f = @(N) 4096 * power(N,10) * exp(-N/3200)-0.05
f_bar = @(N) 4096 * (10*power(N,9)*exp(-N/3200)-power(N,10)*exp(-
N/3200)/3200)
x = 400000

for i = 1:1:20
        new_x = x-  f(x)/f_bar(x);
        fprintf('x = %d\n',new_x);
        if(abs(new_x-x)<1e3)
                break;
        end
        x = new_x;
end
```

```
x = 4.034783e+05
x = 4.069539e+05
x = 4.104270e+05
x = 4.138976e+05
x = 4.173657e+05
x = 4.208313e+05
x = 4.242943e+05
x = 4.277545e+05
x = 4.312109e+05
x = 4.346611e+05
x = 4.380982e+05
x = 4.415040e+05
x = 4.448297e+05
```

x = 4.479518e+05
x = 4.505920e+05
x = 4.523031e+05
x = 4.528986e+05

x = 453000

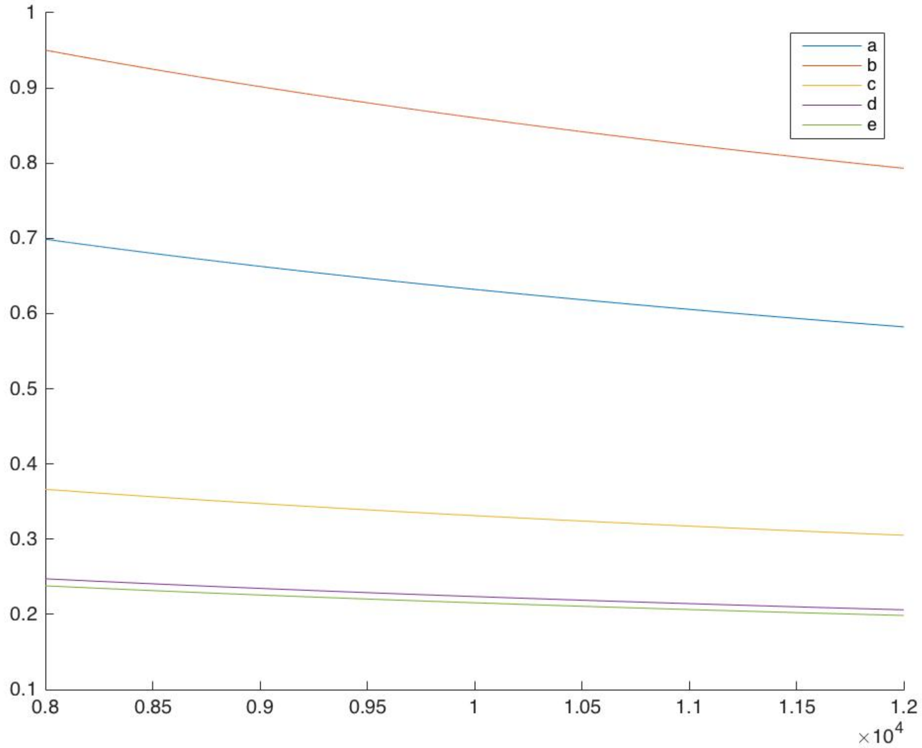
4.

```
dvc = 50
delta = 0.05

mh = @(N) power(N,50)
f1  = @(N) sqrt((8/N)*(50*log(2*N)+log(80)))
f2 = @(N) sqrt((16/N)*(log(2)+50*log(N)-log(0.05)/2))
f3 = @(N) sqrt((2/N)*(log(2)+51*log(N)))+sqrt(2/N+log(20))+1/N
f4 = @(N) 1/N + sqrt(1/power(N,2)+(log(120)+50*log(2*N))/N)
f5 = @(N)  1/(N-2) + sqrt(1/power(N-2,2)+(log(20)+50*log(N))/(N-2))

X = 8000:0.1:12000
Y1 = [];
Y2 = [];
Y3 = [];
Y4 = [];
Y5 = [];
for i = X
        Y1 = [Y1 f1(i)];
        Y2 = [Y2 f2(i)];
        Y3 = [Y3 f3(i)];
        Y4 = [Y4 f4(i)];
        Y5 = [Y5 f5(i)];
end
hold on
plot(X,Y1);
plot(X,Y2);
plot(X,Y3);
plot(X,Y4);
plot(X,Y5);
legend('a','b','c','d','e');
hold off
N = 10000
f1(N) =0.632174915200836
f2(N) =0.860425970706274
f3(N) =2.037707475429844
f4(N) =0.223698293680786
f5(N) =0.215415038524950
```

The tightest bound is given by Devroye



5.

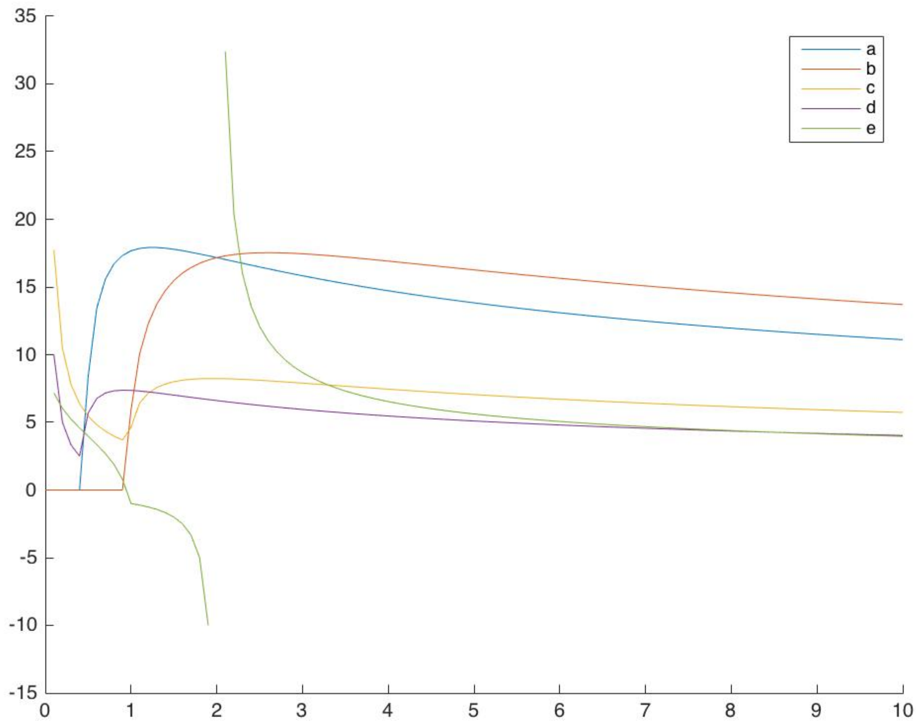
```
dvc = 50
delta = 0.05

mh = @(N) power(N,50)
f1  = @(N) sqrt((8/N)*(50*log(2*N)+log(80)))
f2 = @(N) sqrt((16/N)*(log(2)+50*log(N)-log(0.05)/2))
f3 = @(N) sqrt((2/N)*(log(2)+51*log(N)))+sqrt(2/N+log(20))+1/N
f4 = @(N) 1/N + sqrt(1/power(N,2)+(log(120)+50*log(2*N))/N)
f5 = @(N)  1/(N-2) + sqrt(1/power(N-2,2)+(log(20)+50*log(N))/(N-2))

X = 0:0.1:10;
Y1 = [];
Y2 = [];
Y3 = [];
Y4 = [];
Y5 = [];
for i = X
        Y1 = [Y1 f1(i)];
        Y2 = [Y2 f2(i)];
        Y3 = [Y3 f3(i)];
        Y4 = [Y4 f4(i)];
        Y5 = [Y5 f5(i)];
end
hold on
plot(X,Y1);
```

```
plot(X,Y2);
plot(X,Y3);
plot(X,Y4);
plot(X,Y5);
legend('a','b','c','d','e');
hold off
```

f1(5)= 1.382816e+01
f2(5)= 1.626411e+01
f3(5)= 7.796862e+00
f4(5)= 5.101362e+00
f5(5)= 5.618563e+00
The tighetest bound is given by Parrondo and Van den Broek



6.

2 * (C(N+1,2)-N)+2 = N*(N+1) - 2* N + 2 = N^2 - N +2
For N points, take any 2 of the N+1 points separated by the N points to form a positive interval, minus N of them who chose the right most (W.L.O.G) space, times 2 for the negative intervals, and plus 2 for the all positive and all negative case.

7.

mh(N) = N^2- N +2
N = 1 , mh = 2
N = 2 , mh = 4

N = 3 , mh = 8
N = 4 , mh = 14 < 2^4 = 16
=> VC 3

8.

For any n points, sort them by their distance to the origin.
Choose any 2 of the N+1 space as the two distance required to form the donut, and +1 for the all negative case.
=> C(N+1,2)+1

9.

Given an N dimensional polynomial, there can be at most n roots, spliting the plane into at most N+1 area with alternating signs.
If any two of the neighboring points have the same sign, we can always merge two roots to form double root such that the hypothesis can still generate correct results.
=> dvc >= N+1
For N+2 points with alternating signs, by continuity, there must be n+1 roots => contradiction => dvc < N+2
=> dvc = N+1 for N dimensional polynomial
=> dvc = D+1

10.

2^d

Can generate 2^d hyper-rectangular =>
Each of the 2^d hyper-rectangular regions make decision independently (+1,-1) =>
at most 2^(2^d) labeling of 2^d points => can shatter 2^d => Dvc = 2^d

11.

Given N points located on 4^i ,for i = 1…N, scaling the axis up by 2^(N+2) , we should be able to generate all labeling by some set of alpha_k, that is , when times by 2 with some cautious offset setting, we can alter the sign of the labeling of one single point, and by timing 4 with cautious offsets setting, we can shatter one point. Thus by locating the N points on 4^i, i =1…N, we see that triangle wave can shatter N points. => Triangle Wave can shatter any N, => Dvc = INF.

12.

1<=i<=N-1 (2^i * mh(N-i)) is an upper bound on the growth function mh(N) for n>= dvc >= 2
for N = dvc ,we can see that mh(N) = 2*mh(N-1)

for N = dvc+1 , we can see that mh(N) < 2^N+1 , while 2^i* mh(N-i) = 2^N regardless of the value of i;
for N > dvc+1 , we can see that mh(N) < 2*mh(N-1) (if mh(N) >= 2*mh(N-1) for some N> dvc+1, this means that by adding a single point, we can generate all previous result as well as all possible result of this new data point, the bound of dvc no longer holds =>contradiction)
2^i *mh(N-i) = 2^N for N-i < dvc
for N-i > dvc, or for N > dvc+i , we've shown that mh(N) < 2*mh(N-1) => mh(N) < 2^2*mh(N-2) ...
=> mh(N) <= min(1<=i<=N-1) 2^i mh(N-i) for N >= dvc

13.

Let 2^sqrt(N) be mh(N)
mh(1) <= 2^1
mh(2) = 2 < 2^2 =4
dvc = 1
we know that mh(N) < N^dvc+1
when N = 100
2^sqrt(100) = 1024 < 100^1+1 = 101 contradiction => 2^sqrt(N) cannot be a mh(N)

14.

LHS -> By definition , the VC dimension of empty set is taken as 0
RHS -> The minimum number of points such that every hypothesis in K sets can shatter
< The minimum nuber of points such that one hypothesis in the K sets can shatter
-> dvc(H1^H2^...Hk) < min {(dvc(Hk))} k=1...K

15.

LHS -> The vc dimension of the union of k sets of hypothesis = the maximum number of points that a combination of the k sets of hypothesis can shatter, which should be at least the maximum number of points that a single hypothesis set can shatter
=> max{dvc(Hk)} k=1...K <= dvc(H1UH2...Hk)
RHS -> Suppose dvc(H1) = M , dvc(H2) =N , this means that given M+1 points to H1, it can generate at most 2^M+1 labeling
Also, given N+1 points to H2, it can generate at most 2^N+1 labeling
However, by using a combination of the two hypothesis set H1, H2, given M+N+1 points, the can generate 2^(M+N) * 2 = 2^(M+N+1) of labeling (for example, let H1 in charge of o and H2 in charge x, all lablings of M+N+1 points can be generated)
=> For K sets following the combination rule just mentioned, K-1 intermediate points' labeling can be generated, therefore
dvc(H1UH2...UHK) <= K-1 + (sum(dvc(Hk)) for k=1...K)

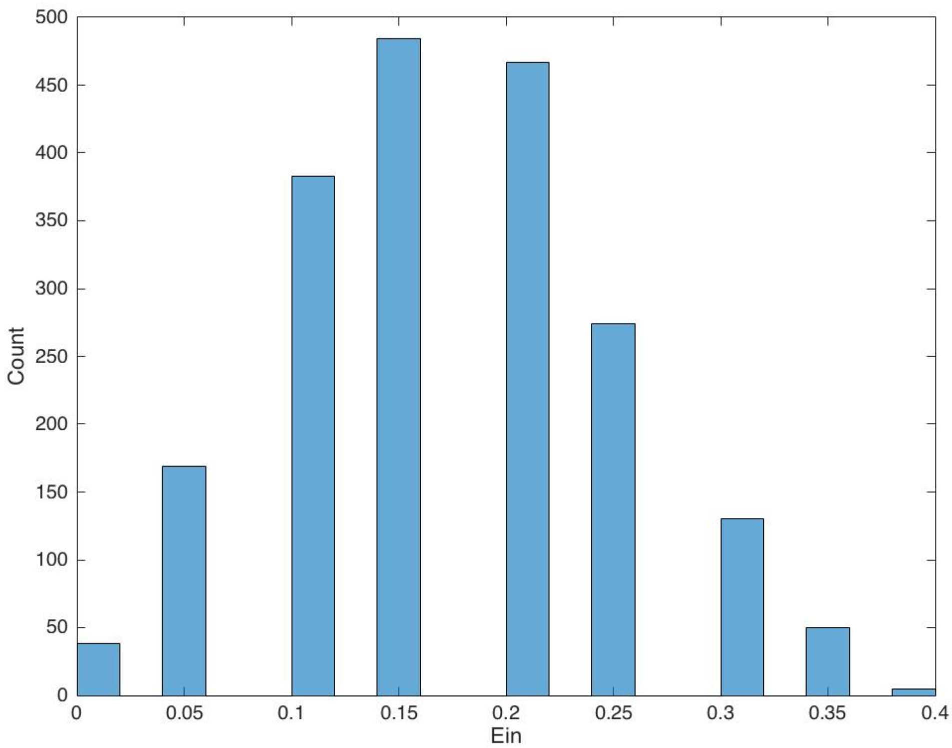
16.

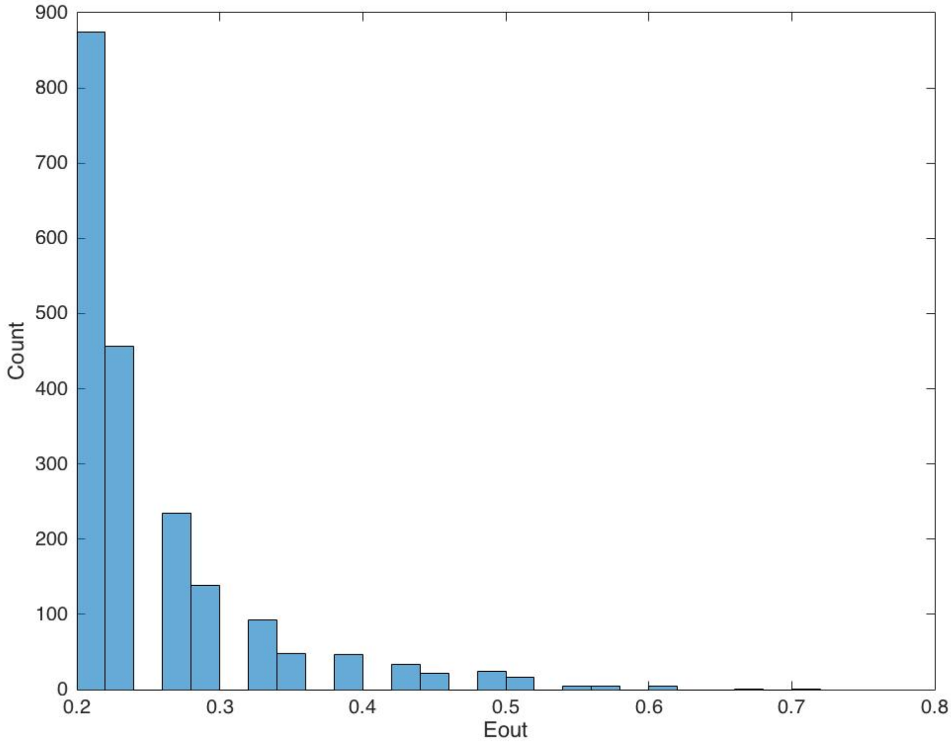
use o denote theta
for s = +1 , o > 0
flip the correct ones ,and maintain the wrong ones
E= (2-|o|)/2 * 0.2 + |o|/2 *0.8
for s = +1 , o < 0
E= (2-|o|)/2 * 0.2 + |o|/2 *0.8
for s = -1 , o >0
E = |o| /2 * 0.2 + (2-|o|)/2*0.8
for s = -1 , o > 0
E = |o| /2 * 0.2 + (2-|o|)/2*0.8

=> E = 0.3|o| +0.2 for s > 0
E = -0.3|o|+0.8 for s< 0
Combine the case of s = +1, -1
E = 0.3*s*|o| + 0.5 + -s*0.3
 = 0.3*s*(|o|-1)+0.5

17. 18.

mean of Ein = 1.719250e-01
mean of Eout = 2.473842e-01





19.

best_index(theta) = 1.773990000000000
best_dim = 4
Ein = 0.025000000000000
direct(s) = -1

20.
Eout = 0.353000000000000