Machine Learning (ENME808E): Homework 3

Koyal Bhartia(116350990)

March 28, 2019

1 Homework 3 Problems

Generalization Error

1.1 Question 1

The modified Hoeffding Inequality provides a way to characterize the generalization error with a probabilistic bound $P[|E_{in}(g)E_{out}(g)| > \epsilon] \leq 2Me^{2\epsilon^2 N}$ for any ϵ ¿ 0. If we set $\epsilon = 0.05$ and want the probability bound $2Me^{2\epsilon^2 N}$ to be at most 0.03, what is the least number of examples N (among the given choices) needed for the case M = 1?

[ANSWER: b]

It is given that the upper limit on the probability limit is 0.03. Thus this can be written as:

$$2Me^{2\epsilon^2 N} \leq 0.03 => e^{2\epsilon^2 N} \leq \frac{0.03}{2M} \tag{1}$$

$=> 2\epsilon^2 N \leq log\frac{0.03}{2M} => N \geq \frac{log\frac{0.03}{2M}}{-2\epsilon^2}$

Plugging in the values for M in the above derivation we get following as the least value for N. For M=1 we get:

$$N \geq \frac{log\frac{0.03}{2}}{-2\epsilon^2} \tag{2}$$

This gives the least value of N as 839.94 which is closest to 1000 from among the given options. Thus the answer is [b]

1.2 Question 2

[ANSWER: b] PLugging in the value of M as 10 in the above equation we get:

$$N \geq \frac{log\frac{0.03}{2*10}}{-2\epsilon^2} \tag{3}$$

The least value of N we get here is: 1300.45. This is closest to 1500 from among the given options. Hence the answer is b i.e 1500

1.3 Question 3

[ANSWER: d] PLugging in the value of M as 100 in the above equation we get:

$$N \geq \frac{log \frac{0.03}{2*100}}{-2\epsilon^2} \tag{4}$$

The least value of N we get here is: 1760.97. This is closest to 2000 from among the given options. Hence the answer is d i.e 2000

Break Point

1.4 Question 4

The (smallest) break point for the Perceptron Model in the two-dimensional case (R^2) is 4 points. What is the smallest break point for the Perceptron Model in R^3 ?

Answer:[B]

Considering 3 fixed points in a 2D plane, we can use the Perceptron model to shatter the points i.e all the 2^3 combinations of +ve and -ve for the points. Further if we take 4 fixed points on the 2D plane and test for shattering of all the possible 2^4 combinations, there exists at least one such pattern which cannot be shattered. Hence the smallest breaking point of the 2D case was 4.

On similar lines, the breaking point of Perceptron Model in the 3D case can be calculated. Here, if we consider shattering of 4 points placed on 2 parallel planes such that in one plane 2 points are diagonally opposite, and similarly 2 other points are diagonally opposite on the parallel plane. Here for all the combinations of +ve and -ve for the 4 given points i.e 2^4 we can easily find a plane which can shatter the points. This was not possible in the case of 2D.

Now, if on the same 2 planes considered above, if we place another point i.e a total of 5 points, we see that there consists cases in the possible 2^5 combinations which cannot be shattered by any plane. Hence in the 2D case the smallest breaking point is 5 i.e option b.

Growth Function

1.5 Question 5

Which of the following are possible formulas for a growth function $m_H(N)$:

Answer:[B]

The growth function is basically used to formalize the effective no of hypotheses. It replaces M in the generalization bound. Hence it is a combinatorial quantity that captures how different the hypotheses in H are and hence how much overlap each event has. It can be proved that the growth function $m_H(N) \leq 2^N$. Also, the possibility of a polynomial as a valid growth function can be tested using the following theorem: If $m_H(N) < 2^k$ for some value k, then

$$m_H(N) \leq \Sigma_{i=0}^{k-1} C_i^N \tag{5}$$

for all N. The RHS is polynomial in N of degree k-1.
Using this theorem to find the possible values of growth functions, we get:

If k=1 we get $m_H(N) \leq C_0^N = 1$
If k=2 we get $m_H(N) \leq C_0^N + C_1^N = 1 + N$
If k=3 we get $m_H(N) \leq C_0^N + C_1^N + C_2^N = 1 + N + C_2^N$

From the conditions and theorem stated above we see that:
Option(i) is a possible formula for a growth function considering a breakpoint of 2.
Similarly Option(ii) is the formula of growth function for a breakpoint of 3. Hence possible.
Option(iii) is not possible because if we take N=1 and plug it in the given option, ie. $\Sigma_{i=1}^{\sqrt{N}} C_i^N$ and n=1 we get C_1^1=1, which means growth function is $m_H(N) = 1$, equating this to 2^k we get breaking point as 0. This is not possible, hence this option is not a valid growth function.
Option(iv) This is similar to the above case. If we plug in N=1 in $2^{N/2}$ we get the growth function as 2^0=1 which means k=0. A breaking point of 0 is not possible, hence $2^{N/2}$ cannot be considered a valid growth function.
Option(v) is the basic condition stated above as the maximum possible value of a growth function. This is the growth function of a convex hull.

Hence option(i),(ii) and (v) are possible. This corresponds to answer option[b].

Fun with Intervals

1.6 Question 6

Answer:[C]

As we know, break point is the value corresponding to the number of points k for which a data set exists, which can't be shattered by the hypothesis set H.

Considering 2 intervals learning model, and testing the shattering of 3 points as follows:
Eg: $+ + -$ All patterns of these 3 points can easily be shattered as there is always a maximum of only 2 intervals, hence it gets shattered for even the worst case which is: $+ - +$ Similarly the shattering of 4 points can be tested as follows: Consider the following 4 points:
$+ - + -$ The above is the worst case among all the 16 possibilities with 4 points which will surely require all least 2 intervals. A 2-interval model will be capable of shattering even the above model of alternate 1's and 0's. Hence 4 cannot be the breakout.
Testing for 5 points on similar lines. Consider an alternate sequence of +1 and -1. $+ - + - .+$ The above sequence clearly requires 3 intervals for the 3 given positions of +. This situation did not occur for the case with 4 datapoints as it was not possible for 3 +vs to be completely separated by -ves to form more than 2 intervals. Hence we see that in the case of 3 alternate +ve we require at least 3 interval hypotheses.

Hence the smallest breaking point of a 2-interval model is 5.

1.7 Question 7

Answer:[C]

Using the theorem for growth function which states that for given N points if K is the breaking point, the growth functions satisfies the following:

$$m_H(N) \le \Sigma_{i=0}^{k-1} C_i^N$$

As already proved in the previous question the breaking point for the 2-interval model is 5. Substitute k=5 in the above equation we get:

$m(H) = C_0^N + C_1^N + C_2^N + C_3^N + C_4^N$

$$m_H(N) = 1 + C_1^N + C_2^N + C_3^N + C_4^N \tag{6}$$

Using the following property: $C_i^{N+1} = C_i^N + C_{i-1}^N$

$C_2^N + C_1^N = C_2^{N+1}$ and $C_4^N + C_3^N = C_4^{N+1}$

Substituing this back into the equation we get the growth function as:

$m_H(N) = 1 + C_2^{N+1} + C_4^{N+1}$ which is option C.

1.8 Question 8

Answer:[D]

Here we try to get the smallest breaking point considering a general case of M-interval learning model.

1-interval:Reflecting back to the problems already solved, we saw that the the breaking point of a 1-interval model was 3. As if we consider the points $+-+$, there are 2 +ves, requiring another interval which is not possible to be shattered by just 1 interval.

2-interval model: Here if we consider an alternate sequence of +ve and -ve, eg: $+-+-+$ we will need at least 3 intervals to shatter it. Hence the breaking point was 5. Also any combination of 4 points i.e $+-+-$ will needs at most 2 intervals which can easily be shattered.

In the same way we had proved that for the 3-interval model, it cannot shatter if there are more than 3 alternately placed +ve. i.e $+-+-+-+$. Hence the shattering point here was 7.

Hence we see that for a case of any no of intervals if the no of alternate +ve is greater that the interval count(M) it cannot shatter. Thus to generate this no of alternate +ve, we need the same no of -ves i.e M.

Hence the breaking point for M interval problem will be M+M+1 i,e 2M+1.

Convex Sets:The Triangle

1.9 Question 9

Answer:[D]

In order to consider a triangle as a learning model where h(x)=+1 if x lies inside the triangle and

-1 otherwise, we can consider the points around the circumference of a circle.

Checking for the shattering of 3 points around the circle It can be clearly observed that with these points, all patterns of +ve and -ve combination of the 3 points can easily be shattered with a triangle. Eg: even $+-+$ can be shattered.

Next, Checking for 5 points along the circumference of circle: Here too like the earlier case we fix 5 points and test for the different combinations of +ve and -ve. Here again all points get successfully shattered. Even for eg: $+-+-+$

Similarly with several trials and errors it is possible to even shatter 7 points along the circumference of a circle. Eg: $+-+-+-+$

Next, when we fix 9 points around the circle, we see that it is not possible to shatter all the possible combinations of +ve and -ve. Eg: $+-+-+-+-+$ cannot be shattered by the triangle learning model.

Thus here the breaking point is 9. Thus the max no of points that be shattered by the hypothesis set is 7.

Non-Convex Sets:Concentric Circles

1.10 Question 10

Answer:[B]

The growth function of any learning model can be obtained by first calculating the breaking point and then computing the growth function using $\Sigma_{i=0}^{k-1} C_i^N$.

Here the 2 concentric circles model of 2 different radius such that it is +1 for $a^2 \leq x_1^2 + x_2^2 \leq b^2$ and -1 otherwise can be considered as a single interval problem. Here the 2 different circles can be considered as the 2 end points.

It has already been calculated above that the breaking point of a 1-interval problem is 3.

Thus using the formula of calculating growth function we get the following:

$\Sigma_{i=0}^{2} C_i^N = C_0^N + C_1^N + C_2^N$

$= 1 + C_1^N + C_2^N$

Now using the property: $C_i^{N+1} = C_i^N + C_{i-1}^N$

Thus $C_1^N + C_2^N$ can be written as C_2^{N+1}

Hence the Growth function is $m_H(N) = 1 + C_2^{N+1}$ i.e option b.

2 Homework 4 Problems

Generalization Error For the next 3 problems, we consider $m_H(N) = N^{d_{VC}} for N > d_{VC}$

2.1 Question 1

Answer:[D]

Given: epsilon=0.05, delta=0.05 $d_{VC} = 10$

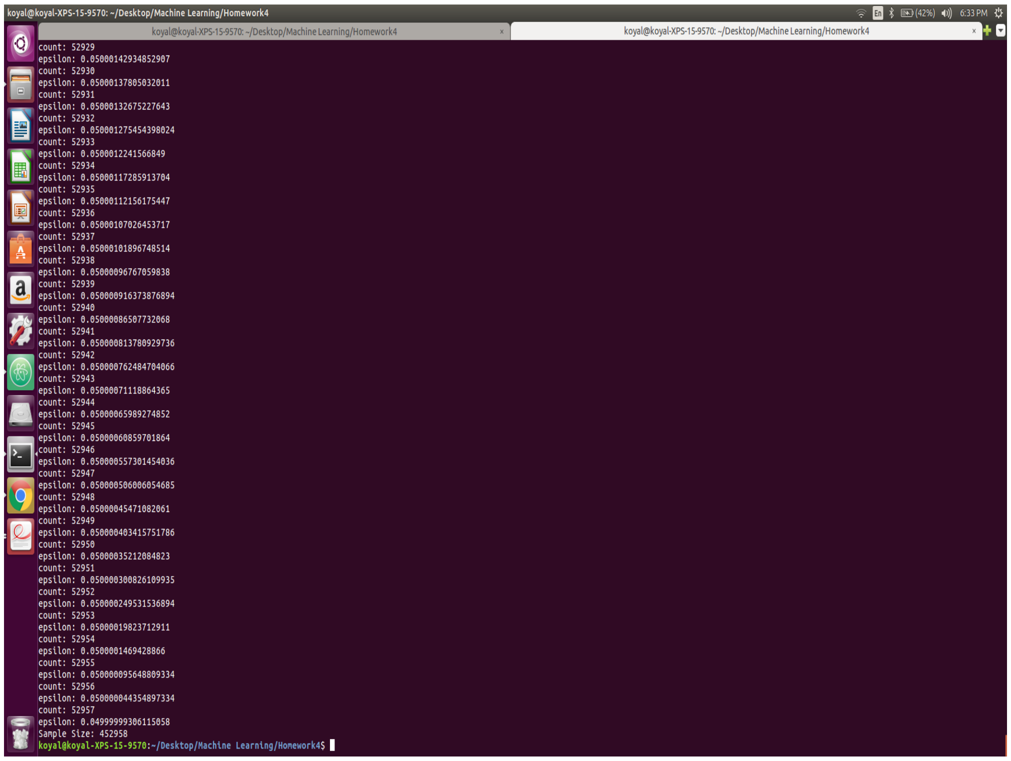
The approximate value of the sample size N is calculate using an iterative process as shown in the Homework 4 - Question 1 python code attached.

Starting with a random value of ϵ as 100 and then repeatedly calculating the value of ϵ using the formula:

$$\epsilon = \sqrt{\frac{8}{N} ln \frac{4m_H(2N)}{\delta}} \tag{7}$$

We get: $N = \frac{8}{\epsilon^2} ln \frac{4m_H(2N)}{\delta}$

Now in each iteration the value of ϵ gets updated and the value of the sample size is incremented. This is continued until ϵ becomes equal to 0.05.



```
koyal@koyal-XPS-15-9570: ~/Desktop/Machine Learning/Homework4
count: 52929
epsilon: 0.05000142934852907
count: 52930
epsilon: 0.05000137805032011
count: 52931
epsilon: 0.05000132675227643
count: 52932
epsilon: 0.050001275454398024
count: 52933
epsilon: 0.0500012241566849
count: 52934
epsilon: 0.05000117285913704
count: 52935
epsilon: 0.05000112156175447
count: 52936
epsilon: 0.05000107026453717
count: 52937
epsilon: 0.05000101896748514
count: 52938
epsilon: 0.05000096767059838
count: 52939
epsilon: 0.050000916373876894
count: 52940
epsilon: 0.05000086507732068
count: 52941
epsilon: 0.050000813780929736
count: 52942
epsilon: 0.050000762484704066
count: 52943
epsilon: 0.05000071118864365
count: 52944
epsilon: 0.05000065989274852
count: 52945
epsilon: 0.05000060859701864
count: 52946
epsilon: 0.050000557301454036
count: 52947
epsilon: 0.050000506006054685
count: 52948
epsilon: 0.05000045471082061
count: 52949
epsilon: 0.050000403415751786
count: 52950
epsilon: 0.05000035212084823
count: 52951
epsilon: 0.050000300826109935
count: 52952
epsilon: 0.050000249531536894
count: 52953
epsilon: 0.05000019823712911
count: 52954
epsilon: 0.0500001469428866
count: 52955
epsilon: 0.050000095648809334
count: 52956
epsilon: 0.050000044354897334
count: 52957
epsilon: 0.04999999306115058
Sample Size: 452958
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning/Homework4$
```

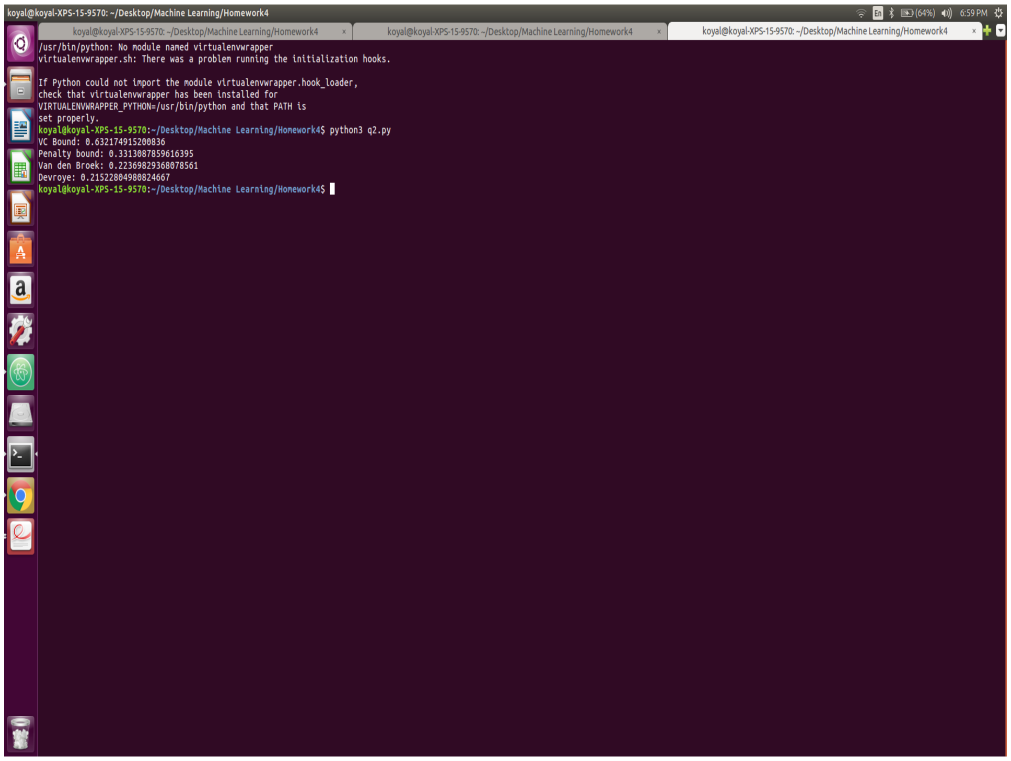
Figure 1: Output of Question 1

As can be seen in the screenshot, after around 52957 iterations, ϵ becomes equal to 0.05 and we get the sample size as **452958.** This is closest to 460000 from among the given options. Hence the answer is d.

2.2 Question 2

Answer:[D] The code of calcualtion this can be found in the codes section. Here we are given the following: $d_{VC} = 10$, delta=0.05 and N=10000.

As the initial value for ϵ is not given, we start with a ramdom value of ϵ as 0, and keep updating the value of ϵ for a count of 10. The given bounds are plotted as a function of N. As seen from the scrrenshot following values of ϵ for each bound is obtained.



```
/usr/bin/python: No module named virtualenvwrapper
virtualenvwrapper.sh: There was a problem running the initialization hooks.

If Python could not import the module virtualenvwrapper.hook_loader,
check that virtualenvwrapper has been installed for
VIRTUALENVWRAPPER_PYTHON=/usr/bin/python and that PATH is
set properly.
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning/Homework4$ python3 q2.py
VC Bound: 0.632174915200836
Penalty bound: 0.3313087859616395
Van den Broek: 0.22369829368078561
Devroye: 0.21522804980824667
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning/Homework4$
```

Figure 2: Output of Question 2

We get following values

VC bound = 0.63217

Penalty bound = 0.3313

Van den Broek = 0.22369

Devroye = 0.2152

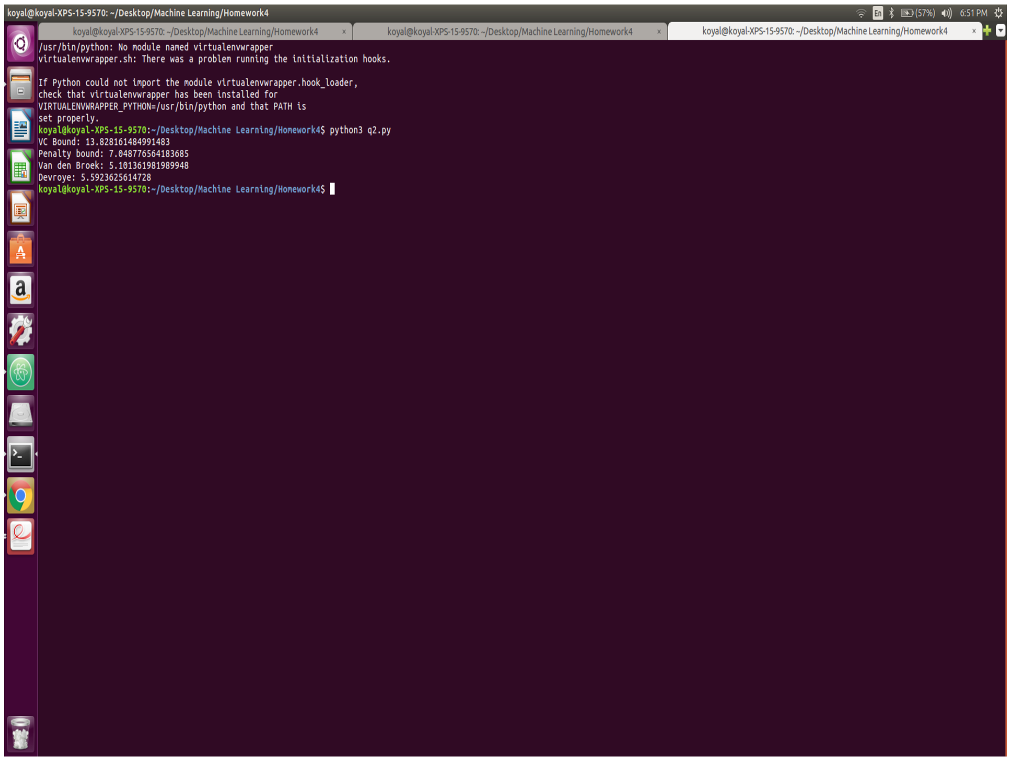
Clearly Devroye is the smallest, hence the answer is option is d.

2.3 Question 3

Answer:[C]

Here all the computation is same as above. The value of N is taken as 5 instead of 10000.

The screenshot shows the values of each of the bounds obtained.



```
/usr/bin/python: No module named virtualenvwrapper
virtualenvwrapper.sh: There was a problem running the initialization hooks.

If Python could not import the module virtualenvwrapper.hook_loader,
check that virtualenvwrapper has been installed for
VIRTUALENVWRAPPER_PYTHON=/usr/bin/python and that PATH is
set properly.
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning/Homework4$ python3 q2.py
VC Bound: 13.828161484991483
Penalty bound: 7.048776564183685
Van den Broek: 5.101361981989948
Devroye: 5.5923625614728
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning/Homework4$
```

Figure 3: Output of Question 3

We get following values

VC bound = 13.828

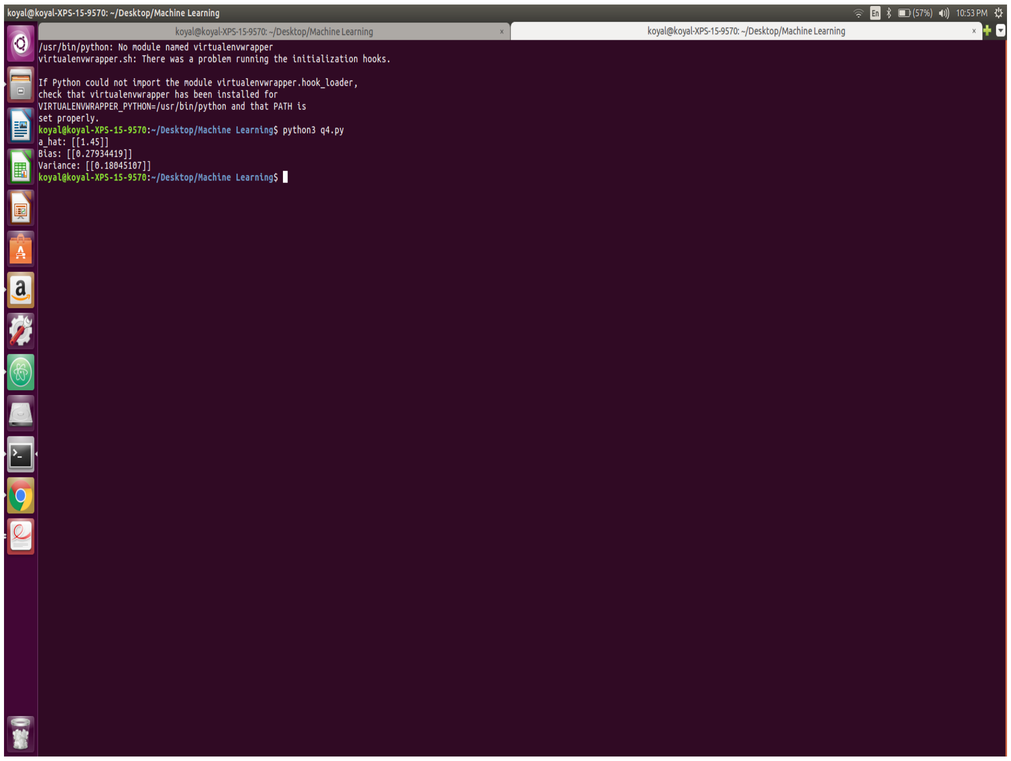
Penalty bound = 7.04877

Van den Broek = 5.10136

Devroye = 5.5923

Clearly Van den has the smallest bound, hence the answer is c.

Bias and Varaince The code for the below questions is attached in the codes section.



```
/usr/bin/python: No module named virtualenvwrapper
virtualenvwrapper.sh: There was a problem running the initialization hooks.

If Python could not import the module virtualenvwrapper.hook_loader,
check that virtualenvwrapper has been installed for
VIRTUALENVWRAPPER_PYTHON=/usr/bin/python and that PATH is
set properly.
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning$ python3 q4.py
a_hat: [[1.45]]
Bias: [[0.27934419]]
Variance: [[0.18045107]]
koyal@koyal-XPS-15-9570:~/Desktop/Machine Learning$
```

Figure 4: Output of Question 4,5,6

The screenshot shows the values of each of the bounds obtained.

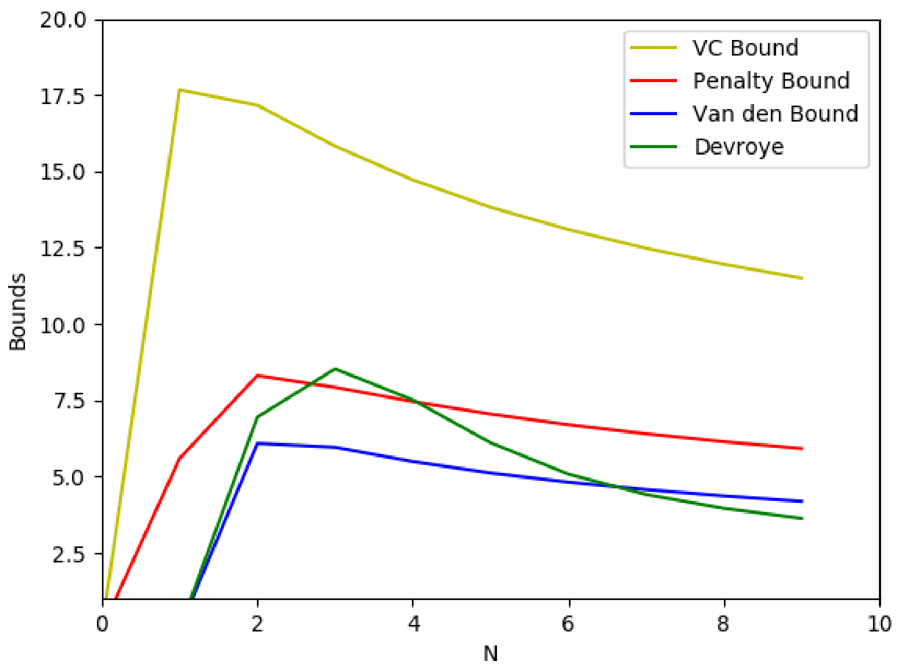


Figure 5: PLot of Bounds

2.4 Question 4

Answer:[E]

To get the expected value of the hypothesis i.e $g_bar(x)$, we take a set of random values varying between -1 to +1. The corresponding value of Y is calculated using the given target function $f(x) = sin(pi*x)$. To calculate the slope generated, we take the pseudo inverse of X i.e the set of randomly generated points and multiply it with Y.

The above is run for about 1000 iterations. The slope of each iteration is saved and the average value is then taken.

As seen from the screeshot we get the average value as 1.45. Thus the expected hypothesis we get is: $g_bar(x) = 1.45x$. This is not given in the options provided, hence the answer is option e.

2.4.1 Question 5

Answer:[B]

Bias is given as $(g_bar(x) - f(x))^2$ Using the expected hypothesis obtained above, we calculate the above difference squared for random values of x. This is repeated for 1000 iterations and then the average is taken. As seen from the screenshot, the average bias obtained is 0.279. This is closest to 0.3 from among the given options.

2.5 Question 6

Answer:[A]

The calculation of variance is done by considering 2 sets of randomly generated numbers. The corresponding Y is found using the target function. Using the pseudo inverse of the random X, we get the slope. This is repaeated 1000 times. Then for any value of x, then average value of the following is calculated over 1000 runs.

$(g(x) - g_bar(x))^2$. As seen in the screenshot we get the value as: 0.180 This is closest to 0.2 from among the given options.

2.6 Question 7

Answer:[B]

We know that $E_{out} = Bias + Varinace$ In the same way that we calculated bias and variance above, we calculate it for each of the given functions, by changing the hypotheses in eacg case and computing the corresponding expected value. The values obtained are: Hypo 1: 0.748100 Hypo 2: 0.476204 Hypo 3: 0.779530 Hypo 4: 0.759747 Hypo 5:1.062782

Hence we see that the least value is obtained for h(x)=ax.

VC Dimesnion

2.7 Question 8

Answer:[C]

Assumption: $m_H(1) = 2$

Equation of growth function is given as: $m_H(N+1) = 2m_H(N) - C_q^N$

It is known that VC dimension is breaking point-1. Hence the above given growth function can be written as:

$$\Sigma_{i=1}^{d_{VC}} C_i^{N+1} = 2\Sigma_{i=1}^{d_{VC}} C_i^N - C_q^N \tag{8}$$

This can be expanded as follows using the property: $C_i^{N+1} = C_i^N + C_{i-1}^N$

$$\Sigma_{i=1}^{d_{VC}} C_i^N + \Sigma_{i=1}^{d_{VC}} C_{i-1}^N = 2\Sigma_{i=1}^{d_{VC}} C_i^N - C_q^N \tag{9}$$

Cancelling $\Sigma_{i=1}^{d_{VC}} C_i^N$ from both sides we get:

$$\Sigma_{i=1}^{d_{VC}} C_{i-1}^N = \Sigma_{i=1}^{d_{VC}} C_i^N - C_q^N \tag{10}$$

which is equal to:

$$\Sigma_{i=1}^{d_{VC}} C_{i-1}^N + C_q^N = \Sigma_{i=1}^{d_{VC}} C_i^N \tag{11}$$

This can be split as follows:

$$\Sigma_{i=1}^{d_{VC}} C_{i-1}^N + C_q^N = \Sigma_{i=1}^{d_{VC}} C_{i-1}^N + C_{d_{VC}}^N \tag{12}$$

Cancelling $\Sigma_{i=1}^{d_{VC}} C_{i-1}^N$ from both sides we get:

$$C_q^N = C_{d_{VC}}^N \tag{13}$$

Hence as can be seen from above, $d_{VC} = q$ Hence the answer is option c.

2.8 Question 9

Answer:[B]

Here we are given K Hypothesis sets. The intersection of the same has to be considered, and the bound on the VC dimension has to be found i.e $d_{VC}(\cap_{k=1}^{K} H_k)$

Clearly here because it is inetrsection, the final hypothses that we consider to find the VC dimension is the hypothses that is common to all the given K hypothesis. And this common hypotheses will obviously be a subset of this hypotheses set. Here the different hypothsis will have different VC dimensions. Now considering only the hypothses which become a part of the intersection subset. Now in this subset there will be a hypothesis which has the **least** breaking point i.e which can shatter at most N points. Hence it's VC dimension will be N. Now because it the intersection we are taking about, the max no of points that can be shattered in the entire set also becomes N. Hence the VC dimension of the subset can at most be the VC dimension of the hypotheses will the least VC dimension. This case is considering that the intersection is not a null set.

In case all the hypothses are completely disjoint. Then the subset will be a empty set. Here as given the VC dimension value can be taken as 0. This is the worst case which gives the min value of the VC dimension.

Hence the inequality that satisfies the value of the dimension of an intersection of hypotheses is: $0 \leq d_{VC}(\cap_{k=1}^{K} H_k) \leq min(d_{VC}(H_k))_{k=1}^{K}$. Hence the answer is option b.

2.9 Question 10

Answer:[E]

The Hypotheses set considered here is the same as that considered above. Here we take the union of all the sets given. Thus here any set will be a subset of the union set. As this union contains completely all the individual hypotheses, the max value of the VC dimension is also a part of this union set. Thus the total VC dimension's lower bound will be the maximum value of the VC dimension from among all the given sets. Thus this forms the lower bound i.e $max(d_{VC}(H_k))_{k=1}^{K}$.

Now if we divide the entire union set into sets of sizes equal to each set's VC dimensions then the total VC dimension can be considered to be a summation of the individual VC dimension of each set i.e $\Sigma_{k=1}^{K} d_{VC}(H_k)$

Now we consider two random hypothesis sets H1 and H2 such that both are: y = d1 + d2 + x. Let the VC dimension of H1 be d1, and that of H2 be d2. Now in the union of H1 and H2 will be able to shatter d1+d2 points. Hence the VC dimension of the union will at least be d1+d2+1. This is due to the overlap between the individual sets. Here 1 can actually be interpreted as K-1 where K is 2 in the H1, H2 case considered. Hence the upper bound on the total Vc dimension will be: $K - 1 + \Sigma_{k=1}^{K} d_{VC}(H_k)$

Hence the total bound can be represeneted as: $max(d_{VC}(H_k))_{k=1}^{K} \leq d_{VC}(\cup_{k=1}^{K}) \leq K - 1 + \Sigma_{k=1}^{K} d_{VC}(H_k)$

Note: Codes for each of the questions is attached below for reference.

3 Codes 1(Generalization Error)

```
#
#   Copyright 2019 Koyal Bhartia
#   @file     H4_Question1.py
#   @author   Koyal Bhartia
#   @version  1.0
#
#   @brief This is the code for Question 1 of Homework 4 from "Learning from Data"
#
# @Description This code calculates the generalization error.
#
#Import statments
import numpy as np
import math
#Using ietrative way to find N
#Let initial value of N=
N=400000
count=0
epsilon=100
while(epsilon >=0.05):
    print("count:",count)
    a=8*np.log(80*math.pow(2*N,10))
    epsilon=math.sqrt(a/N)
    print("epsilon:",epsilon)
    N+=1
    count+=1
print("Sample Size:",N)
```

4 Codes 2(Generalization Error-Bounds)

```
import numpy as np
import math
import matplotlib.pyplot as plt
N=5
delta=0.05
d=50
esp1=np.zeros((10,1))
esp2=np.zeros((10,1))
esp3=np.zeros((10,1))
esp4=np.zeros((10,1))
x=np.zeros((10,1))




for i in range(N-5,N+5):
    x[i,0]=i
    #x=linspace(-N+10,N+10,1000)
    esp1[i,0]=math.sqrt((8/N)*np.log((4*math.pow(2*N,d))/delta))
    #print("VC Bound:",esp1)
    esp2[i,0]=math.sqrt((2/N)*np.log((2*N*math.pow(N,d))))+math.sqrt((2/N)*np.log(1/delta))
    #print("Penalty bound:",esp2)
    count=0
    esp3[N-4,0]=0
    while(count<10):
        esp3[i,0]= np.sqrt((1/N)*(2*esp3[i-1,0]+np.log(6*math.pow(2*N,d)/delta)))
        count+=1
    #print("Van den Broek:",esp3)

    count=0
    esp4[N-4,0]=0
    while(count<10):
        #four= np.sqrt((1/(2*N))*(4*esp*(1+esp)+math.log(4*math.pow(N*N,d)/delta)))
        a=100*math.log(N)+math.log(80)
        b=a+4*esp4[i,0]*(1+esp4[i-1,0])
        c=b/(2*N)
        esp4[i,0]=np.sqrt(c)
        count+=1
    #print("Devroye:",esp4)
print(esp1)
plt.plot(x,esp1,'-y',label='VC Bound')
plt.plot(x,esp2,'-r',label='Penalty Bound')
plt.plot(x,esp3,'-b',label='Van den Bound')
plt.plot(x,esp4,'-g',label='Devroye')
plt.xlabel('N')
plt.ylabel('Bounds')
plt.legend()
plt.show()
```

5 Codes 3(Bias and Variance)

```
import numpy as np
import math
import random as random
import matplotlib.pyplot as plt
m=0
for i in range(1000):
    X1=random.uniform(-1,1)
    X2=random.uniform(-1,1)
    Y1=math.sin(math.pi*X1)
    Y2=math.sin(math.pi*X2)

    X=np.mat([[X1,1],[X2,1]])
    Y=[[Y1],[Y2]]

    value=np.linalg.inv((X.transpose()*X))*X.transpose()*Y

    m=m+abs(value[0])

answer=np.round(m/1000,2)
print("a_hat:",answer)



bias=0
for i in range(1000):
    X1=random.uniform(-1,1)
    bias=bias+(answer*X1-math.sin(math.pi*X1))**2
print("Bias:",bias/1000)




def count(m,slope):
    count=0
    for i in range(1000):
        if (m[i]==slope):
            count=count+1
    return count
m=np.zeros((1000,1))
X1=np.zeros((1000,1))
X2=np.zeros((1000,1))
var=0
for i in range(1000):
    X1[i,0]=random.uniform(-1,1)
    A=X1[i,0]
    X2[i,0]=random.uniform(-1,1)
    B=X2[i,0]
    Y1=math.sin(math.pi*X1[i,0])
    Y2=math.sin(math.pi*X2[i,0])

    X=np.mat([[A,1],[B,1]])
    Y=[[Y1],[Y2]]

    ans=np.linalg.inv((X.transpose()*X))*X.transpose()*Y
    m[i,0]=ans[0,0]

for i in range(1000):
    X_choice1=0.25
    slope=m[i,0]
    var=var+(answer*X_choice1-slope*X_choice1)**2
print("Variance:", var/(1000))
```