HW 4

CS156, Kai Chang

Question 1

Answer Choice: **D**, 460,000
Reasoning: We can use the VC bound to estimat ehte sample complexity for a given learning model. So, fix $\delta > 0$, and we want the generalization error at most ϵ. Then, using our VC generalization bound (Theorem 2.5) $E_{out} \leq E_{in} + \sqrt{\frac{8}{N}\ln\frac{4m_H(2N)}{\delta}}$, and derive our generalization error bound ie. $E_{out} - E_{in} = \epsilon \geq \sqrt{\frac{8}{N}\ln\frac{4m_H(2N)}{\delta}}$. It follows then that $N \geq \frac{8}{\epsilon^2}\ln\frac{4m_H(2N)}{\delta}$.

By substitution, we get $N = \frac{8}{\epsilon^2}\ln\frac{4((2N)^{d_{vc}}+1)}{\delta}$. Thus, a learning model with $d_{vc} = 10$ and a gneralization error at most 0.05 with confidence 95 (ie. $\epsilon = 0.05, \delta = 0.05$), we get (plugging into Wolfram Alpha because Mathematica is too slow).

N ≈ 452,957 over real valued solutions.

Wolfram Alpha code:

- Solve[x == 3199.9999999999995`Log[80.` (1 + 1024 x^10)], x] (because N doesn't work, it believes is Newtons)

Question 2

Answer Choice: **D**, Devroye
Reasoning: As we reach out to larger sample sizes, we first note that the Parrando and Van den Broek has the smallest error. However, when I double check to plug in the values, I find that the Devroye bound (error) actually is the smallest. I think the problem with my plot is that when I convert the implicit function to an explicit function, I lose some of the information by taking only the "real part", ie. the quadratic function solving approach with the inequality really messes something up. Not really sure what the difference is, if you can clarify that would be great.

I took the physical input as the correct answer because that has the least variations. So for the smallest generalization error ϵ (ie. smallest bound)

- Original VC bound: **0.632175**, Code: Sqrt[8/N*Log[4*(2*N)^d/[Delta]]] /. {N -> 10000, d -> 50, [Delta] -> 0.05}
- Rademacher Penalty Bound: **0.331309**, Code: Sqrt[2*Log[2*N*N^d]/N] + *Sqrt[2/N*Log[1/[Delta]]] + 1/N /. {N -> 10000, d -> 50, [Delta] -> 0.05}
- Parrondo and Van den Broek: **0.223698**, Code: Solve[[Epsilon] == Sqrt[1/N (2*[Epsilon] + *Log[6*(2*N)^d/[Delta]])] /. {N -> 10000, d -> 50, [Delta] -> 0.05}, [Epsilon]]
- Devroye: **0.215228**, Code: Solve[[Epsilon] == Sqrt[1/(2*N)*(4*[Epsilon]*(1 + [Epsilon]) + Log[4*(N^2)^d/[Delta]])] /. {N -> 10000, d -> 50, [Delta] -> 0.05}, [Epsilon]]

Question 3

Answer Choice: **C**, Parrondo and Van den Broek
Reasoning: Plugging $N = 5$ into we get the Parrando and Van den Broek bound is smallest. If we look at the plot, the Parrando and Van den Broek dominates and agrees with the numerical input.

The smallest ϵ is (Code is identical to above just N -> 5 instead of 10000)

- Original VC bound: 13.8282
- Rademacher Penalty Bound: 7.04878
- Parrondo and Van den Broek: 5.10136
- Devroye: 5.59313

Hmm, because N = 5, and $N < d_{vc}$, we cannot use that simple approximation bound. Instead, we consider the bound 2^N for the growth function $m_H(N)$, which gives us the values:

- Original VC bound: 4.2546
- Rademacher Penalty Bound: 2.81365
- Parrondo and Van den Broek: 1.74395
- Devroye: 2.26454

Question 4

Answer Choice: **E**, None of the above
Reasoning: see code.

For the mathematical derivation of our linear regression algorithm: consider the cost function being the mean squared error. This is given by such:

$$C = \frac{1}{2}\left((y_2 - ax_2)^2 + (y_1 - ax_1)^2\right)$$

and because we want to minimize this error (or cost) with respect to a, we set the derivative = 0 and solve for the local minima

$$\begin{aligned}\frac{dC}{da} = 0 &= \frac{1}{2}(2(y_2 - ax_2) * -x_2 + 2(y_1 - ax_1) * -x_1)\\ &= \frac{1}{2}(-2x_2(y_2 - ax_2) - 2x_1(y_1 - ax_1))\\ &= -x_2(y_2 - ax_2) - x_1(y_1 - ax_1)\end{aligned}$$

and solving this yields

$$\begin{aligned}-x_2y_2 - x_1y_1 &= -ax_2x_2 - ax_1x_1\\ &= -a(x_2^2 + x_1^2)\end{aligned}$$

which results in $a = \frac{x_1y_1+x_2y_2}{x_1^2+x_2^2}$.

In [89]:
```
import numpy as np
w_list = [] # just [w0, w1] or a + some constant term

def min_lin(x,y):
    '''
    Returns the weight that minimizes the mean squared error. Calculated beforeha
nd.

    Parameters
    ----------
    x : input x
    y : f(x)

    Returns
    -------
    weight (ie. a in h(x) = ax)
    '''
    x1 = x[0]
    x2 = x[1]
    y1 = y[0]
    y2 = y[1]

    return (x1 * y1 + x2 * y2)/(x1**2 + x2**2)

for i in xrange(100000):
    x = np.random.uniform(-1,1, 2)
    y = np.sin(np.pi * x)
    w = min_lin(x,y)
    w_list.append(w)

# import matplotlib.pyplot as plt
# %matplotlib inline

# t = np.arange(-1.0, 1.0, 0.01)
# plt.plot(t, np.sin(np.pi*t))

# for i in xrange(3):
#     x = np.random.uniform(-1,1, 2)
#     y = np.sin(np.pi * x)
#     w = min_lin(x,y)
#     plt.scatter(x,y)
#     plt.plot(x, w*x)


print 'a: ', round(np.mean(w_list, axis=0), 2)
a_ = np.mean(w_list, axis=0)
```

```
a:  1.43
```

Question 5

Answer Choice: **B**, 0.3
Reasoning: see code.

To get the bias, we simply take the average bias of a test set. Remember, the bias is defined as

$$\text{bias}(x) = (\bar{g}(x) - f(x))^2$$

where $\bar{g}$ is a function, the average function, composed of these expected values, and f is the true function. In our case, the average function value is given by $\bar{g}(x) = \sum \frac{h(x)}{N} = a_{avg}$.

```
In [90]: b_list = []
         for i in xrange(10000):
             x = np.random.uniform(-1,1)
             y = np.sin(np.pi * x)
             b_list.append((a_*x - y)**2)

         print 'bias: ', round(np.mean(b_list, axis=0), 2)

         bias:  0.27
```

Question 6

Answer Choice: **A**, 0.2
Reasoning: see code.

The variance is defined as $E_D[(g^{(D)}(x) - \bar{g}(x))^2]$, where $g^{(D)}(x)$ is the $h(x)$ of a single data set (ie. 2 data points), which has a unique and singular a, and $\bar{g}(x)$ is again the average of all of our $h(x)$ giving us the average a. Note that the $g^{(D)}(x)$ value has to be in the list of values we used to arrive at $\bar{g}(x)$.

```
In [91]: v_list = []
         w_list = []
         x_list = [np.random.uniform(-1,1,2) for i in xrange(10000)]
         x_list = np.asarray(x_list)
         y_list = np.sin(np.pi * x_list)

         for i in xrange(len(x_list)):
             w_list.append(min_lin(x_list[i],y_list[i]))

         a_avg = round(np.mean(w_list, axis=0), 2)
         for i in xrange(len(x_list)):
             x1 = x_list[i][0]
             x2 = x_list[i][1]
             w = w_list[i]
             v_list.append((x1*w - x1*a_avg)**2)
             v_list.append((x2*w - x2*a_avg)**2)

         print 'variance: ', round(np.mean(v_list, axis=0), 2)

         variance:  0.2
```

Question 7

Answer Choice: **B**, $h(x) = ax$
Reasoning: The expected value of out-of-sample error is just the bias + the variance. We see from the lecture notes and book that

- Constant model scores a E_{out} of 0.75
- $h(x) = ax$ model gives a score of $0.27 + 0.2 = 0.47$.
- Linear model scores a E_{out} of 1.90.
- Both the quadratic choices can be considered too complex for the size of our data and have a terrible problem of overfitting, so it would have a large out-of-sample error.

Question 8

Answer Choice: **C**, q
Reasoning: The VC dimensions of a hypothesis set H, denoted by d_{vc} is the largest value of N for which $m_H(N) = 2^N$. This means that we are finding the minimal value of N such that our bound breaks.

So, given

$$m_H(N+1) = 2m_H(N) - \binom{N}{q}$$

we know that the bound is such that

$$\begin{aligned} 2^{N+1} &= 2m_H(N) - \binom{N}{q} \\ &= 2 * 2^N - \binom{N}{q} \end{aligned}$$

and it is easy to see that the largest value of N such that we can stay within bounds is such that $N = q - 1$, as $\binom{N}{q} = 0$ if $N < q$. Thus, the minimal value of N such that the bound breaks (the Vapnik-Chervonenkis dimension) is $q - 1 + 1$ (ie. $k = d_{vc} + 1$ or the breakpoint for m_H, we get q.

Remember, $\binom{N}{q} \geq 0$ when $N \geq q$, so $N < q$, and q is our VC dimension.

Question 9

Answer Choice: **B**, *this stuff too much to write*
Reasoning: We will eliminate choices until we end up at the correct result.

Because the VC dimension of an empty set or singleton set is taken as 0, if we consider all possible sets, we can have an intersection of all different hypothesis sets H_i such that we are resulted in an empty or singleton set, so choices *d* and *e* are ruled out because the min of a single set could have a non-zero VC dimension. *Note that {d,e} aren't valid bounds. Only {a,b,c} are valid for the generalized hypothesis sets in this case*.

Now, the sum of the d_{vc} does not make sense as a tight bound because the intersection of the sets should have less than the largest d_{vc}, so summing them up is only a choice to consider if we were to take the union of the sets and all the sets were of different values resulting in an incredible large VC dimension. thus, *a* is ruled out.

So, now it comes down to a matter of min or max, and this is actually relatively straight forward. If we consider the intersection of sets (analogously lets consider vector spaces), it only makes sense that we have the same VC dimension (dimensional spaces) as that of the hypothesis set (vector space) with the smallest VC dimension (dimensional space). The intersection is the dimensions that they share (valid in both hypothesis sets and vector spaces), so the minimum VC dimension hypothesis set could not possibly have dimensions greater than it's VC dimensions that it shares with others because it physically does not exist! (ie. $\mathbb{R}^3$ vector space could not possible have $\mathbb{R}^4$ values intersection because it's only limited to an $\mathbb{R}^3$ space). Thus, *c* is ruled out, leading the only valid solution for the tighest bound to be ***b***.

Question 10

Answer Choice: **E**, too much to write down
Reasoning: Now consider the union of this same generalized scenario! Now, we will use the same reasoning to derive our answer.

We know that *{a,b}* is impossible to have because of the fact that we have positive VC dimensions for evey one of our hypothesis sets, meaning the union at minimum must be 1, in the most extreme scenario where they all have the minimum positive VC dimension and are of the same values. Thus, we can rule *{a,b}* out.

Now, we are left with 3 valid arguments. We can first rule out *c* because the lower bound must be bounded by the maximum VC dimension. Again, because we are dealing with the VC dimension of the union of the hypothesis sets, meaning we take the joint of all the VC dimensions, the minimum VC dimension must be the highest VC dimension among the individual. Consider the analogy of vector spaces, the highest vector space dimension must be the minimum bound because the smaller dimensions can automatically be included in the max bound if they share the same span (ie. $\mathbb{R}^3$ could be in $\mathbb{R}^4$ for a set [*note they are not entire spaces, but rather subspaces*], but the vice versa is false). We can now rule out *c* as the tighest or minimum bound.

So we are left with the two final solutions. Intuition tells me that *e* is the correct answer because in the probabilistic cases with a union bound, we often estimate greater than the sum together. This comes from the quintessential concept of ***total greater than sum of parts***.

Consider the case of 2 1D hypothesis sets, where the $d_{vc} = 2$. Each cannot shatter with 3 points, so $d_{vc} = 2$. now, when we take the union of the two, we can shatter points at 4 (1 on the classification line, 1 not), shatter at 5 (both on the classification line), but we cannot shatter 6 (any tricks to orient our classification line in the 1D line case does not work). *Note, we can consider them as interval ranges for the 1D case, ie. the 1 double-interval in a union of 2 single-interval can work for 5 cases, as having the intervals lie within the 4 decision bounds that switch.* Thus, we get the fact that the d_{vc} is 5, which is NOT just the sum of the two individual d_{vc} but rather coincidentally $K - 1 + \sum d_{vc}$, where $K = 2$, and we get a resulting 5. If we extend this to the 2 2D hypothesis sets, we can see that each $d_{vc} = 3$, but the case of $N = 9$ points causes the classification to fail, and $N = 8$ is the maximum number of point at which this unionized hypothesis set can shatter. This corresponds exactly to the sum of our d_{vc} and the $K - 1$ term which is $3 + 3 + 3 - 1 = 8$.

See scratch work for the two 1D cases for clarification, and potentially 2D case. Also, we choose every point to be opposite sign to adjacent because this creates max possible shattering for the number of points.