CS436/536: Introduction to Machine Learning
Homework 1

Instructions: To solve these problems, you are allowed to consult your classmates, as well as the class textbook (Learning from Data by Abu-Mostafa, Magdon-Ismail, and Lin, which we will call LFD) and the slides posted on Brightspace, but no other sources. You are encouraged to collaborate with other students, while respecting the collaboration policy (please see the module on Academic Honesty on Brightspace). Please write the names of all the other students you collaborated with on the homework. Everyone must write up their assignments separately.

Please write clearly and concisely, and use rigorous, formal arguments. Homework is due at the beginning of lecture, and homework turned in later will be considered late and will use up one of your late days. You must use Brightspace to submit the homework as a single neatly typed pdf file. Hand-drawn formulas or figures are okay and may be included as images within the pdf. If a programming assignment calls for plotting the results, axes must be clearly labeled, and its meaning must be obvious to anyone with only a rudimentary knowledge of machine learning and computer science. Emailed copies will not be accepted.

(1) LFD Exercise 1.7. [100 points]

$\mathbf{x}$	y	g	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0 0 0	○	○	○	○	○	○	○	○	○	○
0 0 1	●	●	●	●	●	●	●	●	●	●
0 1 0	●	●	●	●	●	●	●	●	●	●
0 1 1	○	○	○	○	○	○	○	○	○	○
1 0 0	●	●	●	●	●	●	●	●	●	●
1 0 1		?	○	○	○	○	●	●	●	●
1 1 0		?	○	○	●	●	○	○	●	●
1 1 1		?	○	●	○	●	○	●	○	●

Exercise 1.7

For each of the following learning scenarios in the above problem, evaluate the performance of g on the three points in $\mathcal{X}$ outside $\mathcal{D}$. To measure the performance, compute how many of the 8 possible target functions agree with g on all three points, on two of them, on one of them, and on none of them.

(a) $\mathcal{H}$ has only two hypotheses, one that always returns '●' and one that always returns '○'. The learning algorithm picks the hypothesis that matches the data set the most.

(b) The same $\mathcal{H}$, but the learning algorithm now picks the hypothesis that matches the data set the *least*.

(c) $\mathcal{H} = \{\text{XOR}\}$ (only one hypothesis which is always picked), where XOR is defined by $\text{XOR}(\mathbf{x}) =$ ● if the number of 1's in $\mathbf{x}$ is odd and $\text{XOR}(\mathbf{x}) =$ ○ if the number is even.

(d) $\mathcal{H}$ contains all possible hypotheses (all Boolean functions on three variables), and the learning algorithm picks the hypothesis that agrees with all training examples, but otherwise disagrees the most with the XOR.

Solution :

(a) Here we are given 8 possible target functions and we have test data as shown above in the figure. We need to compute our hypothesis for the out of sample data. Here we are told that there are only 2 Hypotheses case scenarios possible. One where it will always return '•' and the other is where it will always return 'o'.

- Scenario 1('•') i.e it will return (true/1) :

  In this scenario, f(8) matches completely with our hypothesis g, whereas f(4), f(6) and f(7) match for 2 points and the functions f(2), f(3) and f(5) match for exactly 1 point. Function f(1) completely disagrees with our hypothesis.

- Scenario 2('o') i.e it will return (False/0):

  In this scenario, f(1) matches completely with our hypothesis g, whereas f(2), f(3) and f(5) match for 2 points and the functions f(4), f(6) and f(7) match for exactly 1 point. Function f(8) completely disagrees with our hypothesis.

In both the case scenarios we are getting equal output, so we can only deduce that the learning is difficult and we can't choose any 1 hypothesis that matches the data set the most.

(b) Here we have to choose the same hypothesis but now we have to evaluate the hypothesis which matches the least with the data. As in the above example we were not able to evaluate the hypothesis which matches the data set the most because we had an equal number of outputs matching for both the hypotheses. In this case scenario we can consider both the case '•' and 'o'.
For '•', we have function f(1) which matches the least i.e it completely disagrees, and function f(2), f(3) and f(5) disagrees for 2 data points. And functions f(4), f(6) and f(7) disagree for 1 data point only. Function f(8) matches the data the most which is the opposite of what we want.
For 'o', we have function f(8) which matches the least that is it completely disagrees, and function f(4), f(6) and f(7) disagrees for 2 data points. And functions f(2), f(3) and f(5) disagree for 1 data point only. Function f(1) matches the data the most which is the opposite of what we want.
In both case scenarios it becomes difficult to select the hypothesis which matches the data set the least as both the hypothesis have equal output result.

(c) Here we have only 1 hypothesis, which is XOR. This XOR hypothesis will return '•' if we have odd number of 1's or '•' and 'o' if we have even number of 1's or '•'. Seeing the out of sample data set we can see that we have 2 data set having even number of 1's so our hypothesis function XOR will return 'o'. And for the last function we have odd number of 1's so our hypothesis function XOR will return '•' and our hypothesis function XOR will look like the figure beside. This hypothesis function XOR is matching to only 1 function for all the data points i.e to the function f(2), and matching 3 functions for 2 points and matching 3 functions for exactly 1 point and does not match to any point for 1

1 0 1 o
1 1 0 o
1 1 1 •

function. So to conclude our hypothesis function g matches to all the 3 data points for only 1 target function out of 8 target functions.

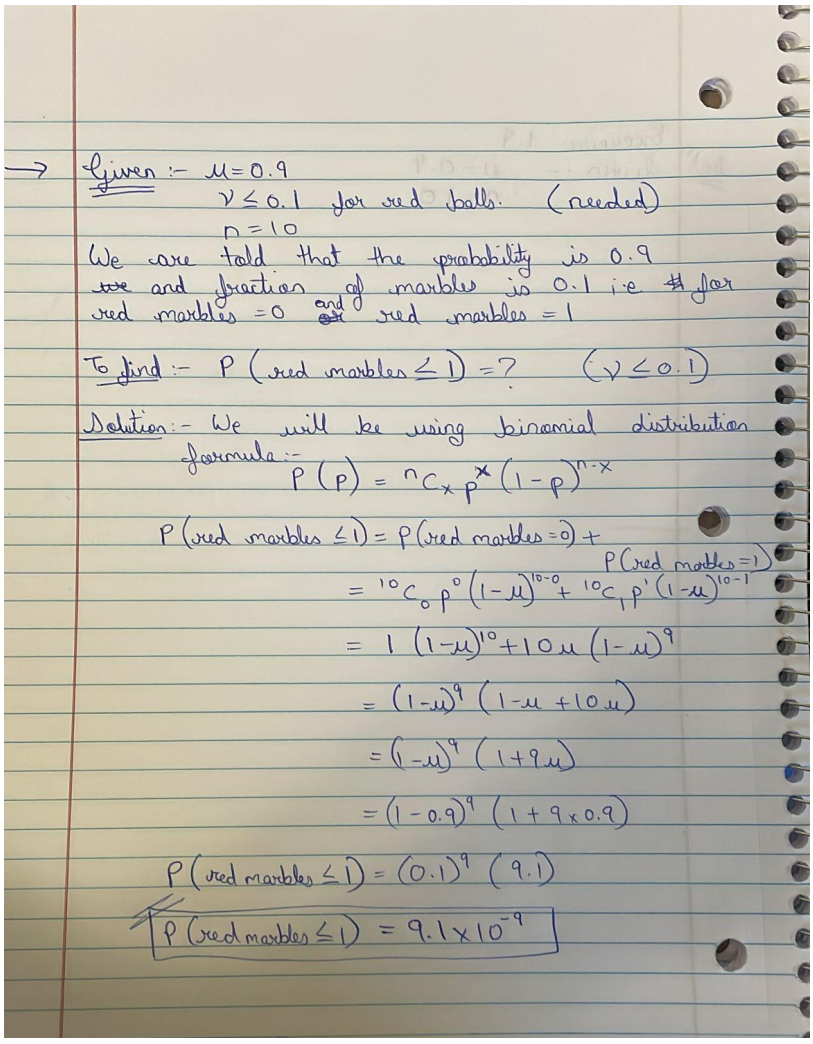
(d) For this we are told that the hypothesis function contains the possible boolean values on 3 variables and that it disagrees the most with the XOR function. So the hypothesis function will look somewhat like the one below.

- This hypothesis function matches the most with only 1 function that is f(7) and
- matches 3 functions for 2 points. For 3 functions it matches for exactly 1 point
- and for 1 function it disagrees completely which is the XOR function from the
- previous question.

(2) LFD Exercise 1.8. [50 points]

Exercise 1.8

If $\mu = 0.9$, what is the probability that a sample of 10 marbles will have $\nu \leq 0.1$? *[Hints: 1. Use binomial distribution. 2. The answer is a very small number.]*



Given :- $\mu = 0.9$

$\nu \leq 0.1$ for red balls. (needed)

$n = 10$

We are told that the probability is 0.9 and fraction of marbles is 0.1 i.e # for red marbles = 0 and red marbles = 1

To find :- $P(\text{red marbles} \leq 1) = ?$ $(\nu \leq 0.1)$

Solution :- We will be using binomial distribution formula :-

$$P(p) = {}^{n}C_{x}\, p^{x} (1-p)^{n-x}$$

$$P(\text{red marbles} \leq 1) = P(\text{red marbles} = 0) + P(\text{red marbles} = 1)$$

$$= {}^{10}C_{0}\, p^{0} (1-\mu)^{10-0} + {}^{10}C_{1}\, p^{1} (1-\mu)^{10-1}$$

$$= 1\,(1-\mu)^{10} + 10\mu\,(1-\mu)^{9}$$

$$= (1-\mu)^{9}\,(1-\mu+10\mu)$$

$$= (1-\mu)^{9}\,(1+9\mu)$$

$$= (1-0.9)^{9}\,(1+9\times 0.9)$$

$$P(\text{red marbles} \leq 1) = (0.1)^{9}\,(9.1)$$

$$\boxed{P(\text{red marbles} \leq 1) = 9.1 \times 10^{-9}}$$

(3) LFD Exercise 1.9. [50 points]

Exercise 1.9

If $\mu = 0.9$, use the Hoeffding Inequality to bound the probability that a sample of 10 marbles will have $\nu \leq 0.1$ and compare the answer to the previous exercise.

Exercise 1.9

Soln Given :- $\mu = 0.9$

$n = 10$

To find :- $P(\nu \leq 0.1) = ?$

Formula :- $P[|\nu - \mu| > \varepsilon] \leq 2e^{-2\varepsilon^2 N}$

Solution :-

$$P(\nu \leq 0.1) = P(|\mu - \nu| > \varepsilon) \leq 2e^{-2\varepsilon^2 N}$$

$$= P(|\mu - \nu| \geq 0.8 \leq P(|\mu - \nu| > 0.7)$$

By keeping $\mu = 0.9$ & $\nu = 0.1$

we get $|\mu - \nu| = 0.8$

So our tolerance should be less than or equal to 0.8.

lets assume our tolerance is 0.7.

$$P(\nu \leq 0.1) = P((|\mu - \nu|) > 0.7) \leq 2e^{-2\varepsilon^2 N}$$

by replacing the values of the variable we get.

$$\leq 2e^{-2(0.7)^2(10)}$$

$$\leq 2e^{-2(0.49)10}$$

$$\leq 2e^{-9.8}$$

$$\leq \frac{2}{e^{9.8}}$$

$$P(\nu \leq 0.1) \approx 0.000110903\ldots$$

(4) LFD Exercise 1.10. [100 points]

Exercise 1.10

Here is an experiment that illustrates the difference between a single bin and multiple bins. Run a computer simulation for flipping 1,000 fair coins. Flip each coin independently 10 times. Let's focus on 3 coins as follows: c_1 is the first coin flipped; c_{rand} is a coin you choose at random; $c_{\min}$ is the coin that had the minimum frequency of heads (pick the earlier one in case of a tie). Let ν_1, ν_{rand} and $\nu_{\min}$ be the fraction of heads you obtain for the respective three coins.

(a) What is μ for the three coins selected?

(b) Repeat this entire experiment a large number of times (e.g., 100,000 runs of the entire experiment) to get several instances of ν_1, ν_{rand} and $\nu_{\min}$ and plot the histograms of the distributions of ν_1, ν_{rand} and $\nu_{\min}$. Notice that which coins end up being c_{rand} and $c_{\min}$ may differ from one run to another.

(c) Using (b), plot estimates for $\mathbb{P}[|\nu-\mu| > \epsilon]$ as a function of ϵ, together with the Hoeffding bound $2e^{-2\epsilon^2 N}$ (on the same graph).

(d) Which coins obey the Hoeffding bound, and which ones do not? Explain why.

(e) Relate part (d) to the multiple bins in Figure 1.10.

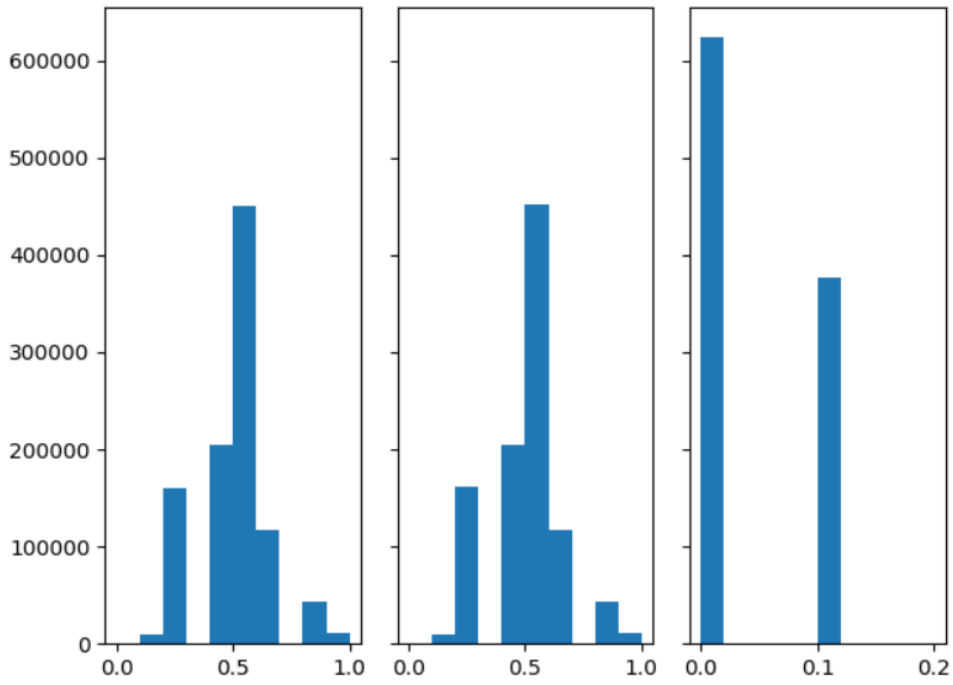
Solution:
Screenshot of the outputs are attached below and the Code is present inside the google colab, please refer to the link below for the google colab.
[Harsimran Singh Dhillon ML_HW1]

(a):

```
The frequency of the first coin v1 =0.7
The frequency of the random coin vrand = 0.4
The frequency of the coin with minimum frequency vmin = 0.1
```

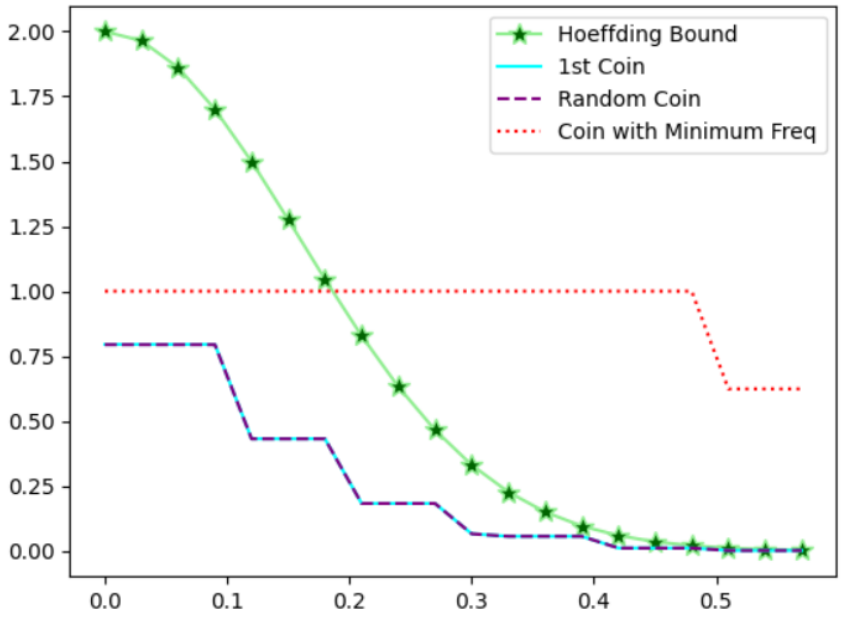
(b):

```
(array([6.23655e+05, 0.00000e+00, 0.00000e+00, 0.00000e+00, 0.00000e+00,
        3.76331e+05, 0.00000e+00, 0.00000e+00, 0.00000e+00, 1.40000e+01]),
 array([0.  , 0.02, 0.04, 0.06, 0.08, 0.1 , 0.12, 0.14, 0.16, 0.18, 0.2 ]),
 <BarContainer object of 10 artists>)
```



(c):

```
<matplotlib.legend.Legend at 0x782865fab550>
```



(d): According to the Hoeffding's inequality condition, the first coin(C1) and the random coin(Crand) obey the Hoeffding bound whereas the coin with the minimum frequency(Cmin) does not obey. This is because the condition says that the hypothesis should be fixed before drawing the samples and to obtain Cmin we have to first draw the samples to calculate the minimum value and then assign it, which violates Hoeffding's inequality condition.

(e): To relate the above condition to multiple bins, we can state that it is like we are choosing a bin with the minimum value after we have sampled all the data in the bins. Which violates our hoeffding bound condition. Whereas for the other two cases (first and random coin) we are choosing the bins before sampling all the data.

(5) LFD Exercise 1.12. [100 points]

Exercise 1.12

A friend comes to you with a learning problem. She says the target function f is *completely* unknown, but she has $4,000$ data points. She is willing to pay you to solve her problem and produce for her a g which approximates f. What is the best that you can promise her among the following:

(a) After learning you will provide her with a g that you will guarantee approximates f well out of sample.

(b) After learning you will provide her with a g, and with high probability the g which you produce will approximate f well out of sample.

(c) One of two things will happen.

(i) You will produce a hypothesis g;

(ii) You will declare that you failed.

If you do return a hypothesis g, then with high probability the g which you produce will approximate f well out of sample.

Answer (c)
Given the condition, that the target function is unknown and we have availability of 4000 data points to produce a hypothesis function g that approximates the target function f. We don't know how complex the target function f is, but as we have a large data set we can state that the probability of our hypothesis function g matching the target function f is pretty high as per the Hoeffding inequality. So to conclude, the answer is Option (c) where we will be able to produce a hypothesis function g with small error and matching our target function to a high extent.

(6) LFD Problem 1.4 (a-e). [300 points]

Problem 1.4 In Exercise 1.4, we use an artificial data set to study the perceptron learning algorithm. This problem leads you to explore the algorithm further with data sets of different sizes and dimensions.

(a) Generate a linearly separable data set of size 20 as indicated in Exercise 1.4. Plot the examples $\{(\mathbf{x}_n, y_n)\}$ as well as the target function f on a plane. Be sure to mark the examples from different classes differently, and add labels to the axes of the plot.

(b) Run the perceptron learning algorithm on the data set above. Report the number of updates that the algorithm takes before converging. Plot the examples $\{(\mathbf{x}_n, y_n)\}$, the target function f, and the final hypothesis g in the same figure. Comment on whether f is close to g.

(c) Repeat everything in (b) with another randomly generated data set of size 20. Compare your results with (b).

(d) Repeat everything in (b) with another randomly generated data set of size 100. Compare your results with (b).

(e) Repeat everything in (b) with another randomly generated data set of size $1,000$. Compare your results with (b).

Answer:

Solved inside the google colab, please refer to the link below for the google colab.

Harsimran Singh Dhillon ML_HW1

(7) LFD Problem 1.7. [300 points]

Problem 1.7 A sample of heads and tails is created by tossing a coin a number of times independently. Assume we have a number of coins that generate different samples independently. For a given coin, let the probability of heads (probability of error) be μ. The probability of obtaining k heads in N tosses of this coin is given by the binomial distribution:

$$P[k \mid N, \mu] = \binom{N}{k} \mu^k (1-\mu)^{N-k}.$$

Remember that the training error ν is $\frac{k}{N}$.

(a) Assume the sample size (N) is 10. If all the coins have $\mu = 0.05$ compute the probability that at least one coin will have $\nu = 0$ for the case of 1 coin, $1,000$ coins, $1,000,000$ coins. Repeat for $\mu = 0.8$.

(b) For the case $N = 6$ and 2 coins with $\mu = 0.5$ for both coins, plot the probability

$$P[\max_i |\nu_i - \mu_i| > \epsilon]$$

for ϵ in the range $[0, 1]$ (the max is over coins). On the same plot show the bound that would be obtained using the Hoeffding Inequality . Remember that for a single coin, the Hoeffding bound is

$$P[|\nu - \mu| > \epsilon] \leq 2e^{-2N\epsilon^2}.$$

[Hint: Use $P[A \text{ or } B] = P[A] + P[B]$ $P[A \text{ and } B] = P[A] + P[B] - P[A]P[B]$, where the last equality follows by independence, to evaluate $P[\max \ldots]$]

Answer:

(a):

Page 1:

Problem 1.7

a) Given :-

$N = 10$

$\mu = 0.05$

To find :- The probability that at least one coin will have $\nu = 0$ i.e $P(\nu = 0)$

Formula :- $P(A) + P(B) = 1$

Solution :- We can manipulate the probability eqn by writing it as.

$P(\nu = 0) + P(\nu > 0) = 1$

As it covers all the case scenarios.

Now Solve it for $P(\nu > 0)$.

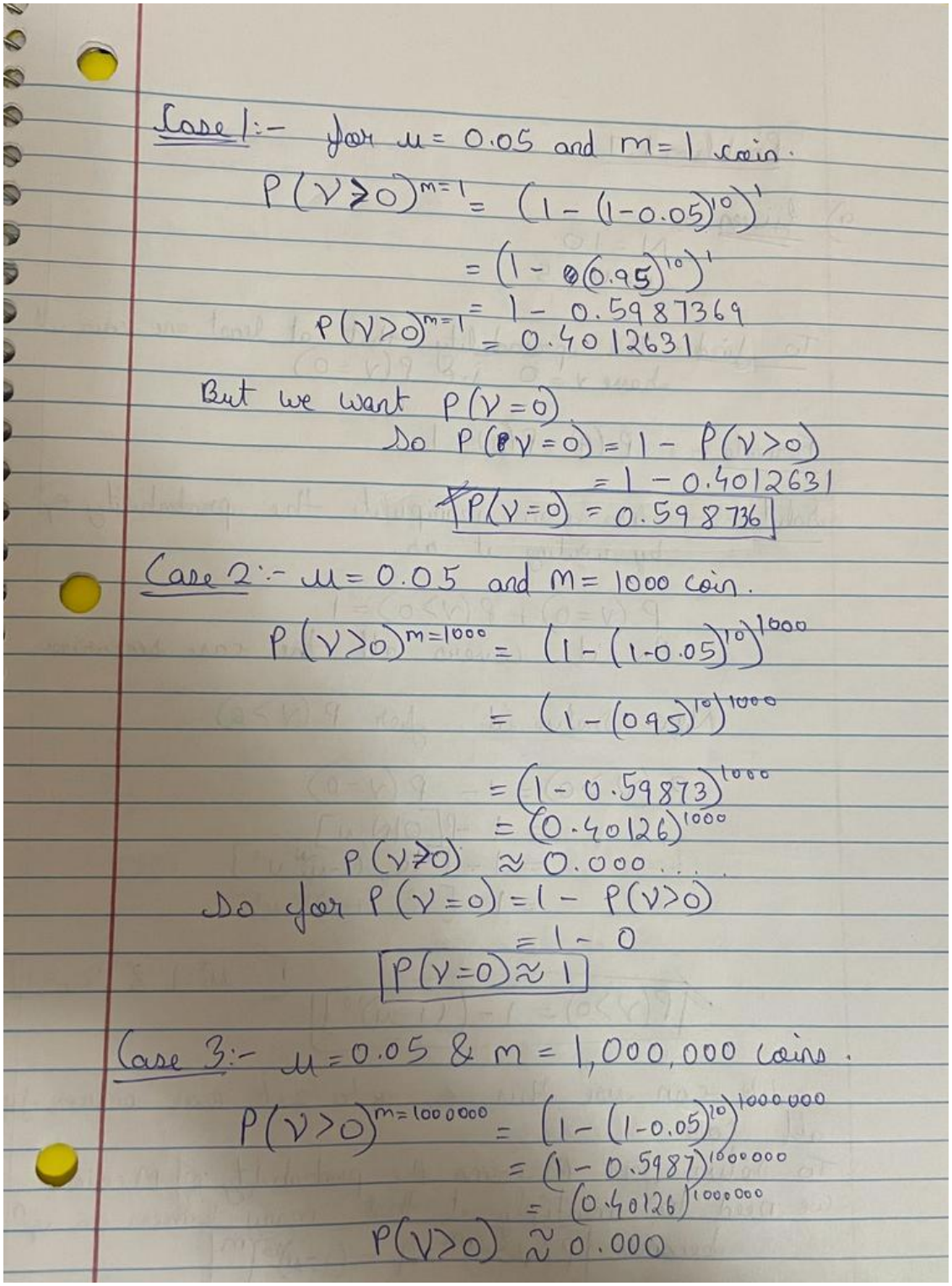
$$P(\nu > 0) = 1 - P(\nu = 0)$$
$$= 1 - P[0|N, \mu]$$
$$= 1 - [{}^{N}C_0 (1-\mu)^N \mu^0]$$
$$= 1 - [{}^{10}C_0 (1-\mu)^{10} \mu^0$$

$$\boxed{P(\nu > 0) = 1 - [(1-\mu)^{10}]} \quad [\because \mu^0 = 1 \ \& \ {}^{10}C_0 = 1]$$

We can use this eq n to get our answer for all case.

To Solve for M coins the probability of M coins we need to multiply it that many times. So eqn becomes $\boxed{P(\nu > 0)^M = [1 - (1-\mu)^{10}]^M}$

Page 2:



Case 1:- for $u = 0.05$ and $m = 1$ coin.

$$P(V \geq 0)^{m=1} = (1 - (1-0.05)^{10})^1$$

$$= (1 - (0.95)^{10})^1$$

$$= 1 - 0.5987369$$

$$P(V>0)^{m=1} = 0.4012631$$

But we want $P(V=0)$

So $P(V=0) = 1 - P(V>0)$

$$= 1 - 0.4012631$$

$$\boxed{P(V=0) = 0.598736}$$

Case 2:- $u = 0.05$ and $m = 1000$ coin.

$$P(V>0)^{m=1000} = (1 - (1-0.05)^{10})^{1000}$$

$$= (1 - (095)^{10})^{1000}$$

$$= (1 - 0.59873)^{1000}$$

$$= (0.40126)^{1000}$$

$$P(V>0) \approx 0.000\ldots$$

So for $P(V=0) = 1 - P(V>0)$

$$= 1 - 0$$

$$\boxed{P(V=0) \approx 1}$$

Case 3:- $u = 0.05$ & $m = 1,000,000$ coins.

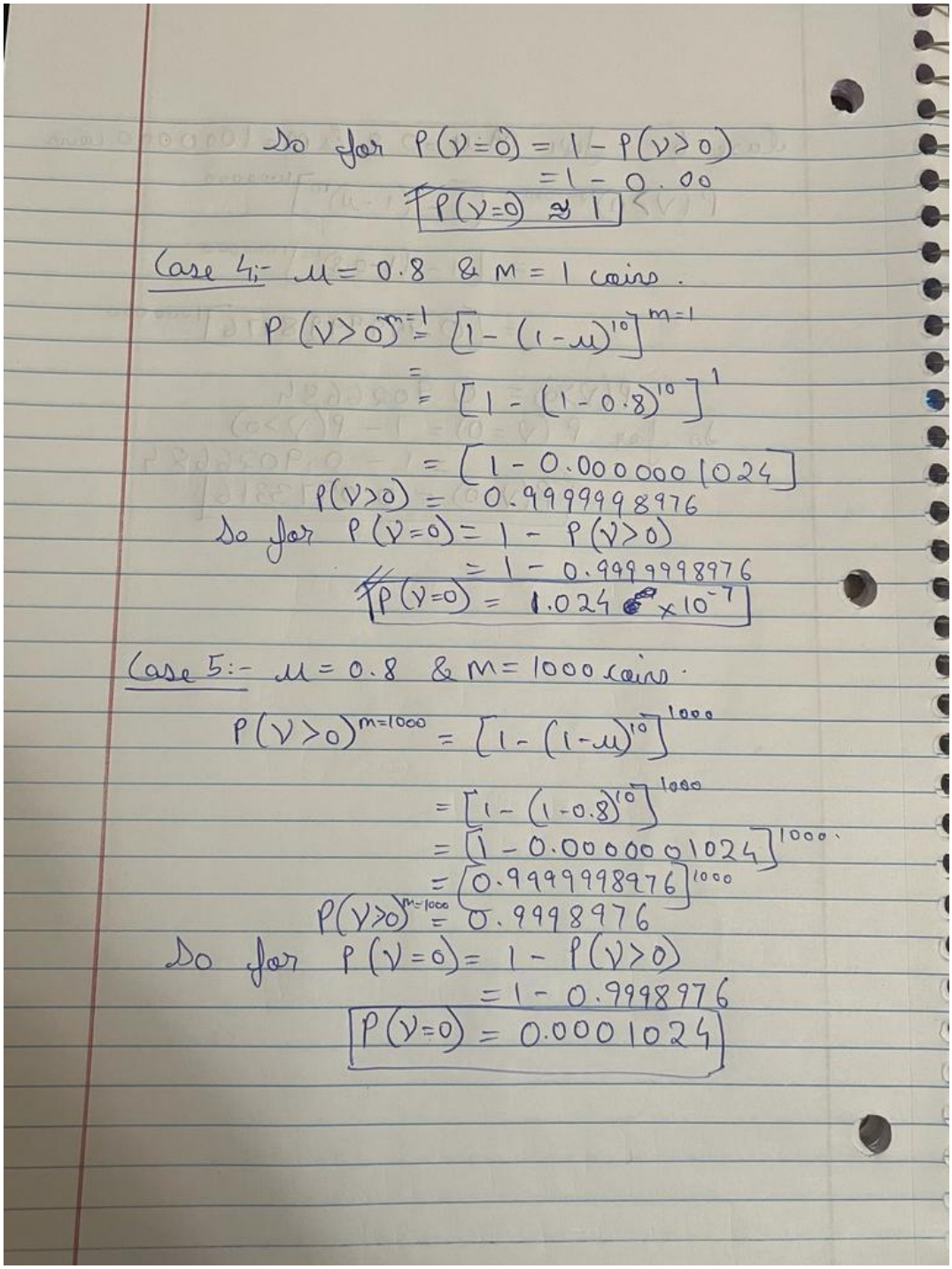
$$P(V>0)^{m=1000000} = (1 - (1-0.05)^{10})^{1000000}$$

$$= (1 - 0.5987)^{1000000}$$

$$= (0.40126)^{1000000}$$

$$P(V>0) \approx 0.000$$

Page 3:



So for $P(Y=0) = 1 - P(Y>0)$

$= 1 - 0.00$

$\boxed{P(Y=0) \approx 1}$

Case 4:- $\mu = 0.8$ & $M = 1$ coins.

$P(Y>0)^{m=1} = [1 - (1-\mu)^{10}]^{m=1}$

$= [1 - (1-0.8)^{10}]^{1}$

$= [1 - 0.0000001024]$

$P(Y>0) = 0.9999998976$

So for $P(Y=0) = 1 - P(Y>0)$

$= 1 - 0.9999998976$

$\boxed{P(Y=0) = 1.024 \times 10^{-7}}$

Case 5:- $\mu = 0.8$ & $M = 1000$ coins.

$P(Y>0)^{m=1000} = [1 - (1-\mu)^{10}]^{1000}$

$= [1 - (1-0.8)^{10}]^{1000}$

$= [1 - 0.0000001024]^{1000}$

$= [0.9999998976]^{1000}$

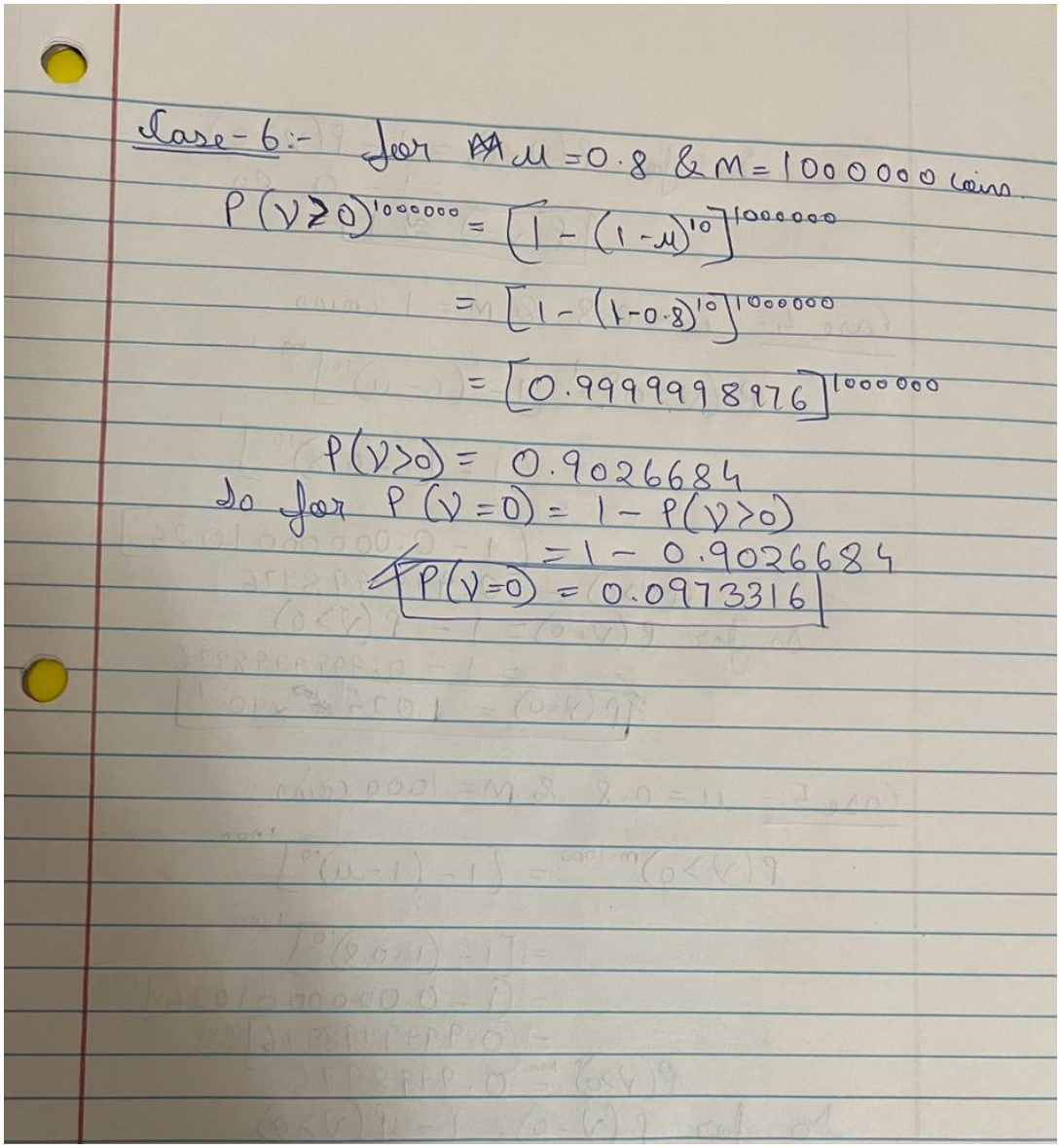
$P(Y>0)^{m=1000} = 0.9998976$

So for $P(Y=0) = 1 - P(Y>0)$

$= 1 - 0.9998976$

$\boxed{P(Y=0) = 0.0001024}$

Page 4:



(b) : Solved inside the google colab, please refer to the link below for the google colab.

Harsimran Singh Dhillon ML_HW1

Access Link

Please refer to the link below for the google colab.

Harsimran Singh Dhillon ML_HW1